

# **Differentiated Services On IBM 221x Routers**

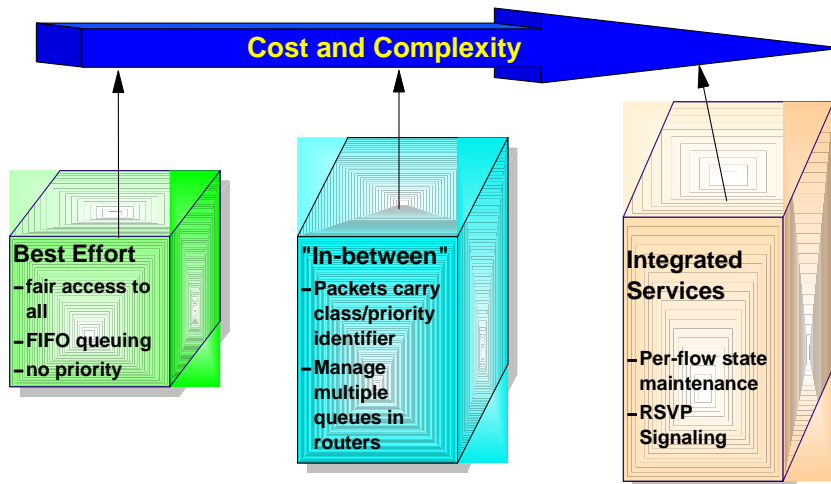
Document Owner : Lynda Linney  
Document Last Edited : 01/08/99, 17:08

*The aim of this document is to give the reader an introduction to Differentiated Services (DiffServ). This was first shipped in the 2210, 2212 & 2216 in V3.3 of their code. The enhancements to DiffServ being shipped in V3.4 are also covered in this document.*

- ✓ Aims to provide scalable service differentiation in the Internet
  - ✓ Implemented in 2210, 2212, 2216 & Network Utility
- ✗ What is wrong with IP today?
  - ✗ Best-effort is often insufficient
  - ✗ Can not meet application requirements, such as delay or bandwidth sensitivities
  - ✗ Difficult to give priority service to specific applications
  - ✗ Rising bandwidth costs
  - ✗ Difficult for network providers to offer and deliver differing service level agreements

The aim of this presentation is to give an understanding of the Differentiated Services architecture as it relates to IPv4 networks and how it is implemented in IBM routers.

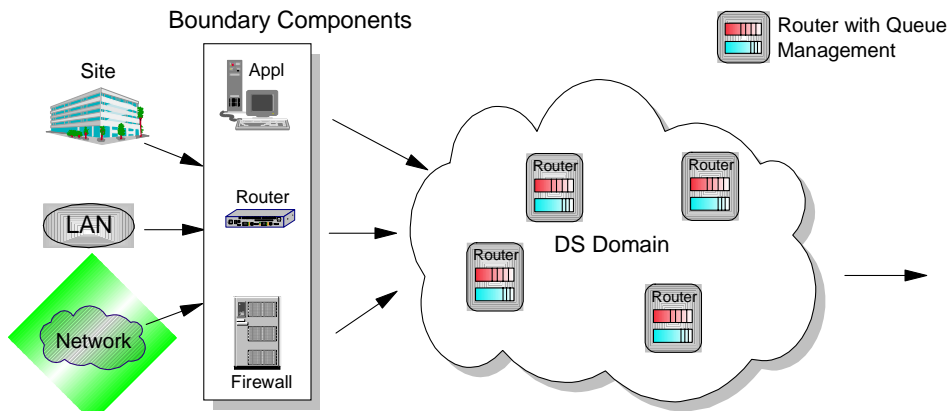
The architecture is also defined for IPv6, but as IBM routers currently only implement it for IPv4, IPv6 will not be discussed in detail. For details of how it is applied to IPv6, the reader is pointed in the direction of the RFCs and IETF drafts - see references at the end of this presentation.



Most forwarding devices installed in an IP network today work on a "best efforts" basis. So the first packet in is likely to be the first packet out of the output interface. There is little to no priority given to traffic. Although in the IPv4 header there is a byte called the Type of Service (TOS) byte, very few vendor's products implemented priority based on this field.

As applications have evolved, much effort has been focused on the issue of how to get priority, quality of service (QoS) into an IP network. Resource Reservation Protocol (RSVP) was developed to allow applications to signal the requirements of the traffic flow. This allows the receiver to request the necessary resources in the intervening network. This approach requires each intervening device to reserve the resources for this traffic flow and keep state information for the flow. The RSVP signalling flows between the source and receivers of the traffic. This approach does not scale well into large networks, and in particular is not regarded as a viable methodology for QoS in the Internet.

Differentiated services (DiffServ or DS) lies between the extremes of best effort, with no priority, and RSVP, with a large overhead. DiffServ has lower overhead than RSVP because it requires no signalling; instead the priority of the packet is transmitted in the IP header. It provides prioritisation where best effort does not, and it formalises the requirement for implementing queuing mechanisms in the network.



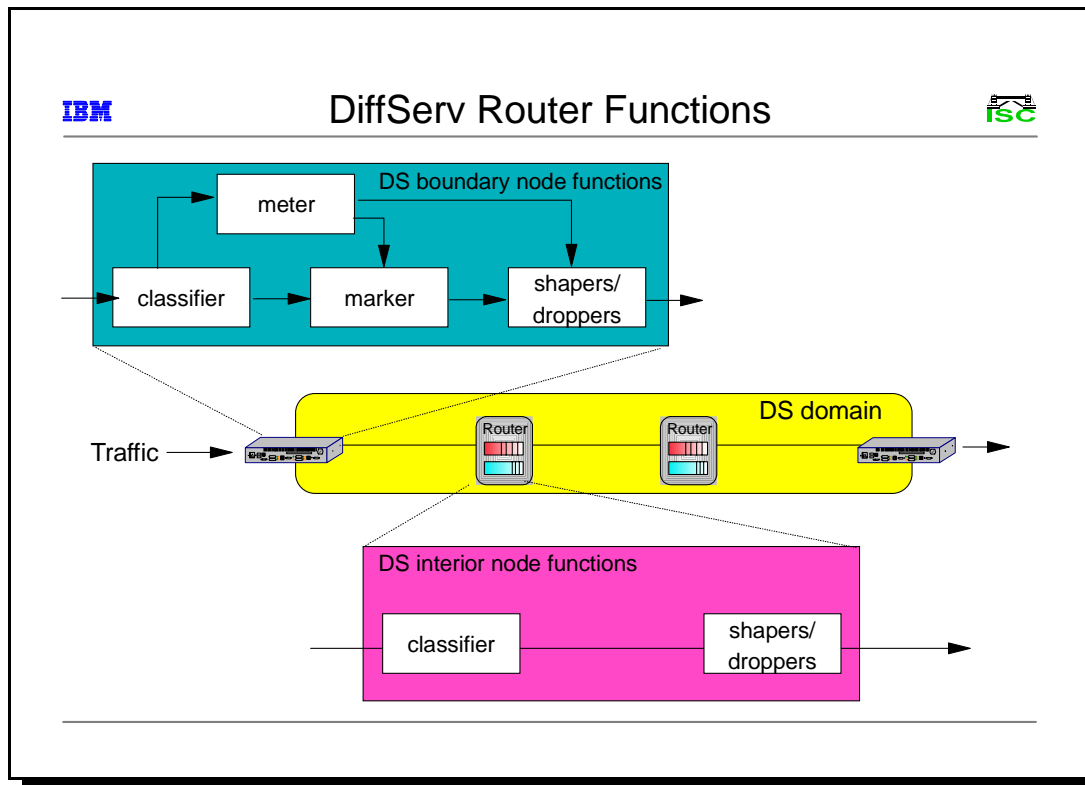
- Edge components classify and mark bits in packet header based on service policy
- Packets serviced at each router based on bits in header

The diagram above shows where the key components of DiffServ networks are deployed. DiffServ defines the concept of a DS domain, which is a contiguous set of DS nodes that apply a common set of policies to the traffic flowing through the domain. A DS domain has DS boundary nodes and DS interior nodes.

DS boundary nodes are responsible for classifying and possibly conditioning traffic that is entering the DS domain. A DS boundary node can either connect two DS domains or connect a DS domain to a non-DS capable domain. For traffic entering the domain, the policy defines which traffic needs a differentiated service, defines how that traffic should be marked by the DS boundary node and how the traffic should be handled.

DS interior nodes, such as routers, inspect the classification of the packets and act according to the defined policies.

Even from an overview of the role of the DS boundary and interior nodes, it becomes clear that DS will be more scalable than IntServ. Complex classification and conditioning functions are performed at the network boundary; and depending on the policies defined, traffic flows with similar requirements are aggregated. This means that per-application flows or per-customers flow states are not maintained. This will become clearer as we look at how classification is achieved and how policies are defined.



This diagram highlights the differences between the functions of boundary nodes and interior nodes in a DS domain.

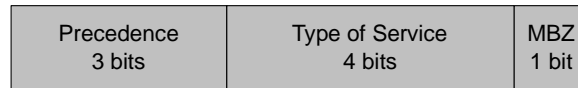
The DiffServ architecture defines two important components in a DS node - the classifier and traffic conditioning components. These functions are more complex in the DS boundary node than in the interior node. Both interior and boundary nodes have a classifier. There are two types of classifiers, one that classifies the traffic stream based on the DS classification only and one that looks at multiple fields in the IP header. These classifiers are known as behaviour aggregate (BA) and multifield (MF) classifiers respectively. The DS classification marking is known as the DS codepoint that will be discussed on the next page.

If the marker was acting as a BA classifier, a DS policy would define that traffic entering the DS domain with DSCP  $x$  should either be handled in a specific fashion, or be mapped to a different classification with a new set of handling definitions. An interior node typically has a BA classifier - it looks at the classification and handles the packet as defined in the policy. A boundary node can also have a BA classifier; typically this scenario would be implemented when the boundary node is at the boundary of two different DS domains. MF classifiers inspect a combination of one or more of the fields in the IP header, such as IP source and destination address, port and protocol, to determine the handling requirements. MF classifiers are only located in DS boundary nodes.

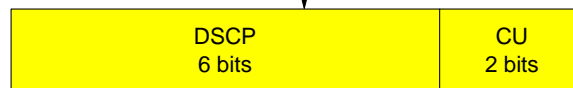
The other key component is the traffic conditioners. The architecture defines:

- *meters* that are used to measure the traffic stream against the traffic profile. It passes state information to other conditioning functions to trigger a particular action for each packet. The action may depend on whether the packet is in-profile or out-of-profile.
- *markers* set the DS codepoint
- *shapers* delay some or all of the packets in the stream to ensure it complies to the traffic profile.
- *droppers* discard some or all of packets in a stream to ensure compliance to the profile. This is often referred to as policing the stream.

Typically, meters and markers are only found in the DS boundary node, whereas shapers and droppers are performed by every forwarding device in the traffic's path.



Redefined



DSCP are mapped to Per Hop Behaviours (PHBs)

Up to now we have just said that the traffic is classified - how is this classification indicated? The IPv4 TOS byte has been redefined as the DS field. The original IPv4 TOS byte was defined in RFC791 along the abstract parameters of precedence, delay, throughput and reliability. Over the years, the definition of the TOS byte has been expanded and most router vendors conform to RFC1812 - Requirements for IPv4 Routers.

The precedence bits of the TOS byte denote the relative importance of a packet. For example precedence bits 111 imply that the packet is network control, 000 implies the packet is routine, 001 priority, etc. - for a full listing refer to the RFC. The indication of delay, throughput or reliability requirements is shown in the TOS field of the TOS octet. A setting of 1000 in the TOS field inferred that the packet should experience minimum delay; 0100 should have its throughput maximised and 0010 should have maximised reliability.

The RFC also defined that "hosts and routers should consider the value of the TOS byte when choosing an appropriate path to get the datagram to its destination." The ability to do TOS routing was originally defined as part of the OSPF v2 standard. As few vendors had implemented it, it was removed in the latest OSPF v2 RFC.

DiffServ reuses the TOS byte; i.e. DS has redefined the fields as shown in the slide. The DSCP is the DS codepoint and CU is currently unused. The DS field has some backward compatibility with the TOS byte as defined in RFC1812 as described in RFC2474. The DiffServ architecture defines the concept of per-hop behaviours (PHBs) which is "a description of the externally observable forwarding behaviour." A PHB is really the definition of what special forwarding treatment is required by packets marked by this codepoint.

- **Default PHB**
  - ▶ best efforts traffic
  - ▶ recommended 000000
- **Class selector**
  - ▶ for backward compatibility to RFC1812
  - ▶ codepoints xxx000
- **Expedited forwarding PHB - RFC2598**
  - ▶ used by premium traffic
  - ▶ recommended codepoint :101110
- **Assured Forwarding PHB - RFC2597**
  - ▶ four independently forwarded classes

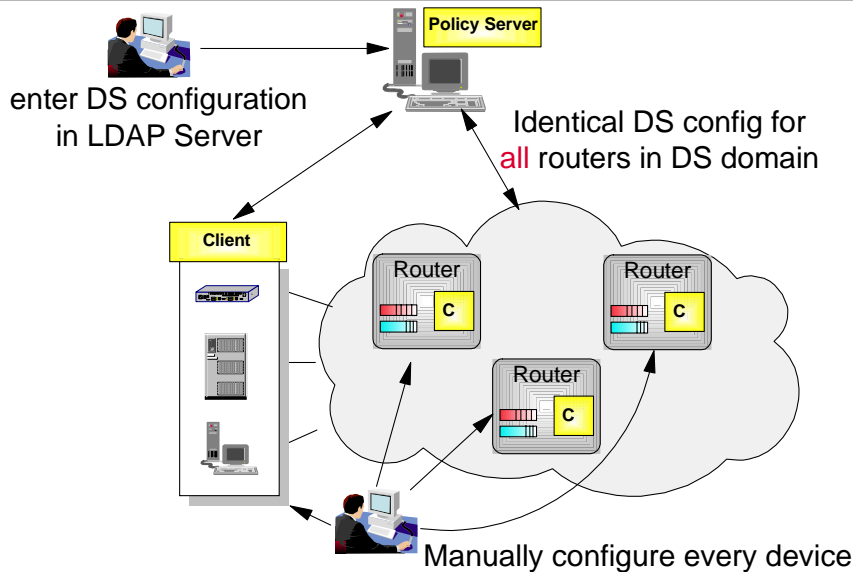
	Class 1	Class 2	Class 3	Class 4
Low drop precedence	001010	010010	011010	100010
Medium drop precedence	001100	010100	011100	100100
High drop precedence	001110	010110	011100	100110

RFC2474 "Definitions of the DS field in IPv4 and IPv6 headers" defines two sets of codepoints, one for the default PHB and one for backward compatibility.

The default PHB is for best efforts traffic. This ensures backwards compatibility with the best effort forwarding existing in RFC1812 routers today. To achieve backwards compatibility, without sacrificing future flexibility, class selector codepoints have been defined. The DSCP is xxx000 and it should yield at least two independently forwarded classes. The PHB of traffic with a higher codepoint should get preferential forwarding over that with a lower value. For example 11x000 would get higher priority over 000000. This maintains the usage of 110 and 111 precedence settings for routing traffic.

There are two other significant PHBs : Expedited Forwarding (EF) PHB which is document in RFC2598 and Assured Forwarding (AF) PHB which is in RFC2597. EF can be used by traffic that has low loss, low latency, low jitter, assured bandwidth end-to-end service requirements. This is often referred to as premium service, or a virtual leased line. Low loss, etc., is achieved by ensuring that the aggregate sees no (or very small) queues. Implementations must provide a means to limit the damage EF traffic can inflict on other traffic (e.g. via token bucket). Devices at the edge of a DS domain must police all EF marked packets to the rate defined. Packets in excess of the rate must be dropped.

AF aims to provide delivery of IP packets in four independently forwarded classes, each class has three drop precedencies. In case of congestion, the drop precedence of a packet determines the relative importance of the packet within the AF class. A congested DS node tries to protect packets with a lower drop precedence value from being lost by preferably discarding packets with a higher drop precedence value. The first three bits of the codepoint identifies the class, 001, 010, 011 and 100 and the last three bits define the drop precedence - 010 for low drop precedence (i.e. drop last), 100 for medium drop precedence and 110 for high drop precedence (i.e. dropped first).



In this presentation we have talked about the "policy" of the DS domain. A policy needs to specify

1. How can the traffic be identified? i.e. how to classify it
2. What resources do classified packets require?

Let's take the requirement that all traffic going from managing director's workstation to the finance department should be subject to minimal delay. A policy would define that traffic going from the IP address of the director's workstation to the IP address(es) of the finance department should be classified with a particular DS codepoint and should receive expedited forwarding. The policy would also define that expedited forwarding requires 10% of the bandwidth and buffers.

Either all or some of this information is required by every node in the DS domain. The configuration could be manually entered into each device in the network, by whatever configuration methods are available. It can also be conceived that SNMP could be used to send the configuration to each device in turn.

Another approach is to have a policy server which can be accessed by clients in the network. One such approach is use the of LDAP. This requires that all the DS-capable devices need to be an LDAP client. LDAP servers can hold (amongst other items) DiffServ polices. That means that all DS nodes at the boundary of the DS-region could be defined to retrieve the same DiffServ policies from the LDAP server. A minimal configuration would contain such details of what traffic should be marked, how it should be marked and define the handling requirements. All interior DS nodes would retrieve polices that state that packets with a certain codepoint require specific handling requirements. This approach also ensures consistency of definitions throughout the network. See separate policy based network presentation for more details.

- Compare to
  1. priority marking e.g. IPv4 precedence
    - refinement
  2. service marking e.g. IPv4 TOS bits
    - DS does not affect routing
    - TOS bits very generic and does not accommodate growth
  3. label switching e.g. FR, ATM, MPLS
    - forwarding states need to be maintained
  4. Integrated Services/RSVP
    - requires signalling
    - does not scale unless aggregation is achievable
    - but DS can be used to aggregate RSVP flows!

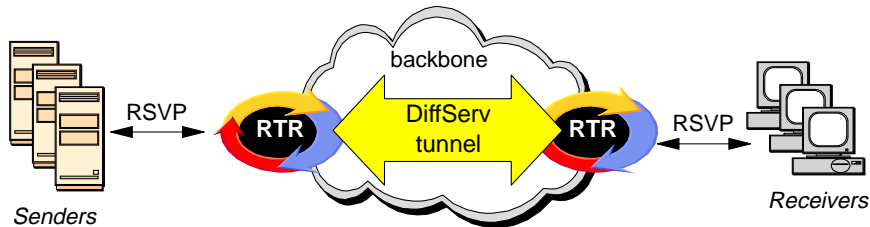
We have seen that DiffServ redefines the IPv4 TOS byte. Is it really that different from existing models? These models include relative priority marking, service marking, label switching, Integrated Services/RSVP, and static per-hop classification.

Examples of the relative priority marking model include IPv4 Precedence marking, 802.5 Token Ring priority, and the default interpretation of 802.1p traffic classes. In this model a relative priority or precedence is selected for the packet. DiffServ can be considered a refinement of this model, since it specifies the role of boundary nodes and traffic conditioners. The PHB model permits more general forwarding behaviours than delay or discard priority.

An example of service marking is the IPv4 TOS in which each packet is marked with a request for a "type of service", such as "minimise delay", "maximise throughput", etc. Network nodes may select routing paths or forwarding behaviours that satisfy the request. This is different from DiffServ. The DS field is not used to control route selection. The TOS markings are very generic and do not easily accommodate growth for future services and would involve configuration of the TOS-to-forwarding-behaviour associations in each core network node.

Examples of the label switching model include Frame Relay, ATM, and MPLS. In this model each hop along a network path defines path forwarding states and traffic management. Traffic aggregates are associated with a label switched path at an ingress node, are marked with a forwarding label that is used to lookup the nexthop node, the per-hop forwarding behaviour, and the replacement label at each hop. This model permits fine granularity of resource allocation to traffic streams, but comes at the cost of additional management and configuration requirements to establish and maintain the label switched paths.

The Integrated Services/RSVP model relies upon traditional datagram forwarding, but allows sources and receivers to exchange signalling messages that establish additional packet classifications and forwarding states on each node along the path between them. In the absence of state aggregation, the amount of state on each node scales in proportion to the number of concurrent reservations. This model also requires application support for the RSVP signalling protocol.



RSVP services mapped to DiffServ services  
at boundary router

It is accepted that RSVP will not scale into large backbones. In RFC2208 it states "...techniques are being developed that will, at the "edge" of the backbone, aggregate together the streams that require special treatment. Within the backbone, various less costly approaches would then be used to set aside resources for the aggregate as a whole."

One such proposal looks at aggregation on two different types of backbone - one that is RSVP-capable backbone and one that is not. If the backbone is RSVP capable, it is proposed that flows could be aggregated into an RSVP tunnel that traverses the backbone. Another proposed method of aggregation is to map RSVP traffic to a specific value in the TOS field (or DS codepoint). The value of the field could indicate the service class (controlled load or guaranteed service) and whether the packet was in- or out-of-profile.

It is the opinion of many people that "integrated services (IntServ) and DiffServ are complementary tools in the pursuit of end-to-end QoS. Each serves an important purpose". End-to-end QoS can be achieved between two IntServ regions that are separated by a DiffServ domain. At the boundary of the IntServ network, the IntServ service type is mapped to a specific DS-codepoint. PATH and RESV messages would be handled as normal in the IntServ regions, whilst being carried transparently across the DiffServ region with the defined DS codepoint.

Whilst the concept of mapping the IntServ to DiffServ is easy, there are several issues - how to hide the RSVP messages from the backbone routers whilst still allowing the boundary routers to identify the messages, how to locate the boundary routers, how to update the PATH messages at the egress routers. For more details on these problems, and the solutions proposed see the appropriate IETF drafts cited in the bibliography.

In IBM routers at this time, the PATH messages are not hidden so they will be handled by any routers that are RSVP capable in the DiffServ backbone.

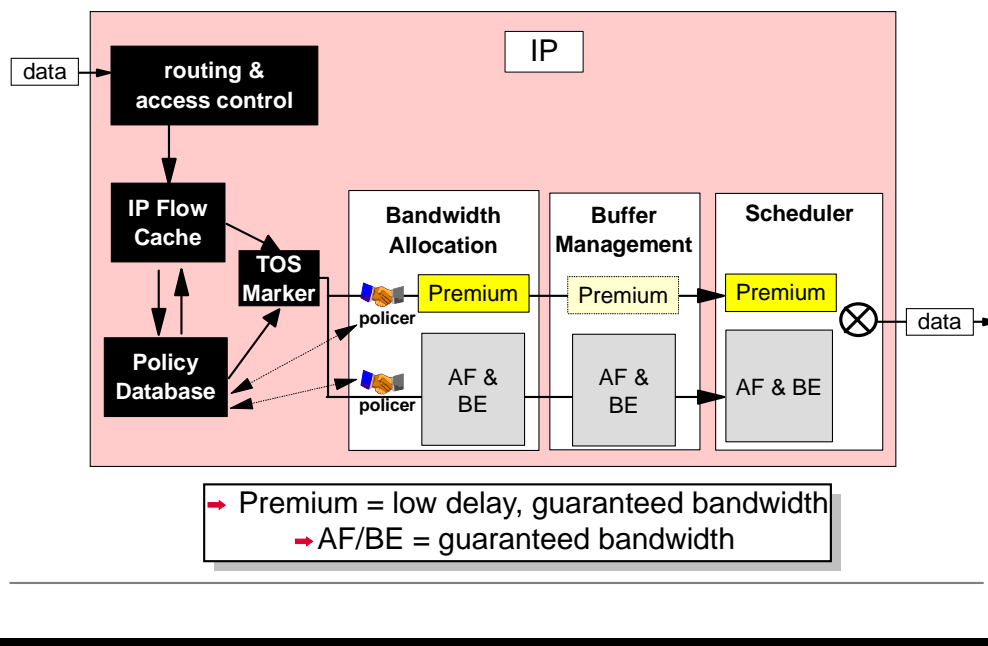
---

## What's Implemented in 2210, 2212, 2216 & Network Utility?

---

Up to now we have focused on the DiffServ architecture. We will now look at what has been implemented on IBM routers since V3.3 and look at what is new in V3.4. DiffServ can also be thought of as a "Service Level Agreements Phase II", a continuation of SLA Phase I that was delivered in V3.2 of the routers' code. Phase I delivered the ability to:

- filter IP packets based on their TOS byte
- modify the TOS byte
- do route selection based on the TOS byte
- assign traffic to BRS classes based on the TOS byte.



This diagram represents the IP and DiffServ components that are within an IBM 221x routers. DiffServ is implemented as a combination of buffer allocation, buffer management and the scheduler. DiffServ is supported on egress PPP, multilink PPP (new in v3.4) and frame relay interfaces. The router has two queues - one for handling premium traffic and another that handles assured forwarding (AF) and best effort (BE) traffic. A stream is created to aggregate individual flows and corresponds to a particular DiffServ policy object. Streams are assigned buffer space and bandwidth.

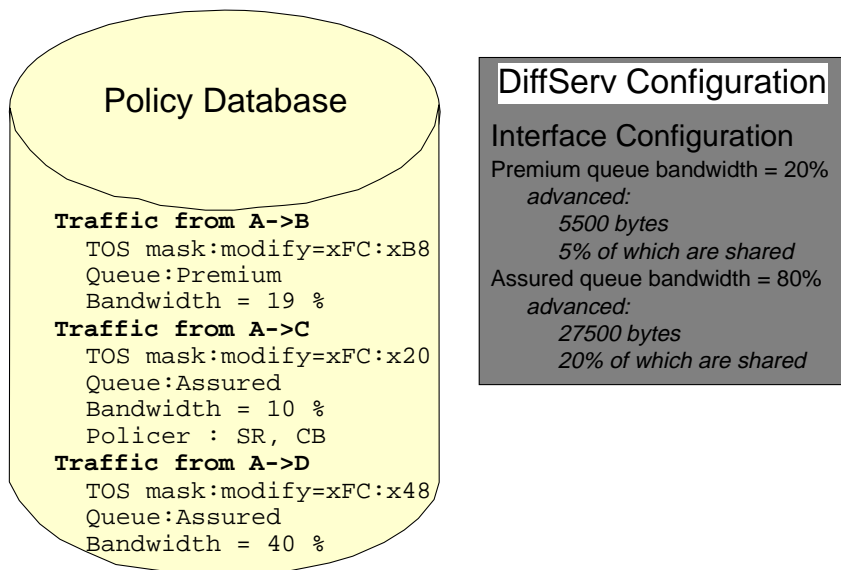
Traffic assigned to the premium queue is guaranteed bandwidth and low delays. This queue is policed to ensure that traffic rate does not exceed that which was defined in the policy. The premium queue will give similar handling characteristics to those defined by the expedited forwarding PHB group. It is a recommendation that only one or two streams use this queue - we will see why later.

Traffic assigned to the AF/BE queue is only guaranteed bandwidth. New in V3.4 is the ability to police the AF/BE queue. Traffic that is not assigned to either queue is handled as best effort. IP network control traffic, such as OSPF, is handled by the AF/BE queue.

When an IP packet is received on an inbound interface, it will pass through the input filters, global access controls, outbound filters and then reach the IP flow cache. If the traffic flow has been handled previously it will have an entry in the IP flow cache. The cache entry will indicate how this packet is to be marked, the associated stream ID, which, in turn, specifies which queue this stream is handled by.

If there is no entry in the cache, the router looks in its policy database to determine what handling the traffic requires. The router can look at the source and destination IP address, the protocol number, the source and destination port numbers, and the current settings of the TOS byte/DS-codepoint to determine if special handling is required. If a policy is defined for the traffic flow it will state which queue the traffic is to be handled by, define how much bandwidth this flow is entitled to and how the packet should be marked. This information is also passed to the IP flow cache.

The router now knows how to mark the packet, so lets look at the bandwidth allocation in detail.



DiffServ is defined in two places - from the DS configuration the characteristics of the DS interface are defined and the policy database stores the policy information for the network. In the policy database shown above there are three policies defined. One from traffic between A and B which is going to be handled by the premium queue and requires 19% of the output bandwidth. Traffic from A to C is going to be placed in the AF queue and is going to be policed via a single rate colour blind marker - more on this later. Traffic from A to D will also be handled by the AF queue, but is not policed.

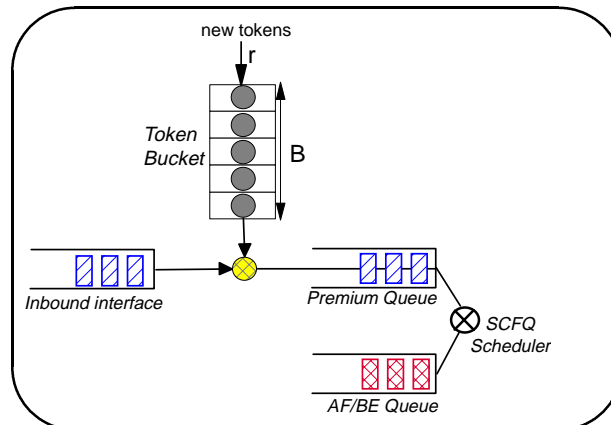
In DS configuration shown on the right is the default values. By default, the premium queue is assigned 20% of the bandwidth. Of this 95% can be allocated to streams requiring low delays and guaranteed bandwidth. The extra 5% of the premium queue's bandwidth is held in a shared pool that can be used by all premium queue streams. This means that the sum of the bandwidth requirements for policies which assign traffic to premium queue cannot exceed  $(0.95 \times 20\%) = 19\%$  of the output bandwidth. If you have one policy that defines a need for 10% of the bandwidth and another policy that requires 10% of the bandwidth, traffic from the first stream to send will get placed in premium queue's buffers and the second stream's traffic will be handled as best efforts traffic because only one of the streams will find enough resources to be initiated.

The AF/BE queue is assigned 80% of the output bandwidth, and of this 80% can be assigned to streams requiring guaranteed bandwidth. This leaves  $(0.80 \times 80\%) = 64\%$  for QoS allocations. By default best efforts and control traffic get 10% and 5% of the output bandwidth respectively. Therefore, the sum of the bandwidth requirements of policies using the AF/BE queue cannot exceed about 50% of the output bandwidth.

The diagram above shows that the router has been configured with a policy that states that traffic going from A to B should be placed in the premium queue, get 19% of the bandwidth and be marked with setting 0xB8. This traffic will occupy all of the buffers assigned to the premium queue. The premium queue has 5500 bytes of buffers, 95% of which are divided between the streams as defined by the policies. The extra 5% can be used by traffic which has a requirement for it. All streams within a queue are allowed an equal share of the buffers, so the portion a stream can use is  $1/N$  where N is the number of active streams.

Our policy defines 19% of the output bandwidth. If the traffic flow is less than 19% of the bandwidth the policer plays no role. If the traffic flow from A to B exceeds the policy and sends traffic which requires more than 19% of the bandwidth, the policer drop packets. The policer uses a leaky bucket regulator. This is a mechanism to control the rate at which data is sent to the network. A leaky bucket approach allows short bursts of traffic into

the network but enforces the average delivery rate over time. The next diagram shows the concept of leaky bucket regulator:



There are two key parameters - the token rate  $r$  and the bucket size  $B$ . Tokens are added to the token bucket at the rate of  $r$ . When the bucket contains  $B$  tokens, no more tokens will be added. For each transmitted packet, tokens are removed from the bucket at the rate of one token for each byte transmitted. If there are no tokens in the bucket the packet is dropped.

So let's take an example where the traffic requires 19% of the bandwidth on a 2 Mbps link. This means that the rate cannot exceed 38,912 bytes per second. By default,  $r$  is 38,912 bytes per second. When the router receives the first packet which requires handling via the premium queue, it calculates the rate  $r$  (38,912 in this example) and defines the bucket size  $B$ . The default size of  $B$  is 2200 bytes. So if the traffic has a burst of less than 2200 bytes and the token bucket is full, then the packet will make it through.

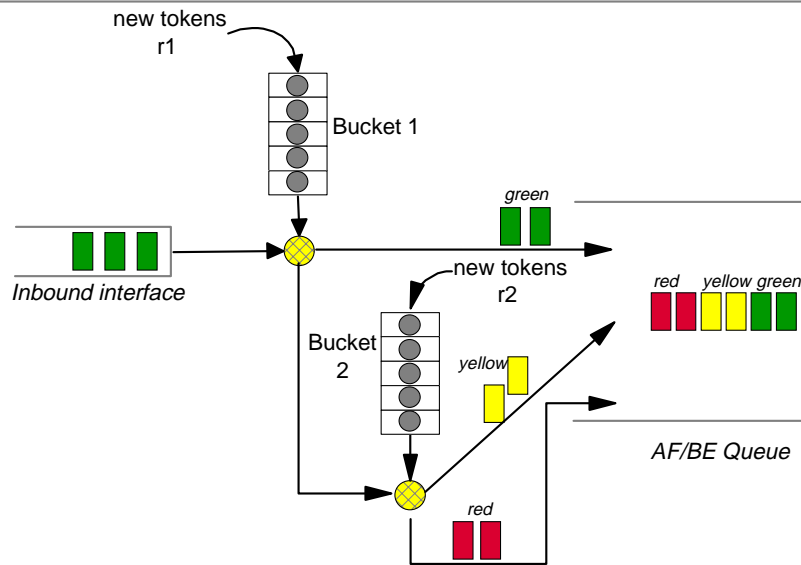
If the user's traffic rate continually exceeds the policy, it will be using tokens at a faster rate than they are being replenished. Eventually there will be no tokens available, and packets will be dropped. Dropped packets are shown as "policed" in the output from the **list streams** command.

In summary, if the packets are less than 2200 bytes and arrive in evenly spaced intervals so that the average rate is less than 38,912 bytes per second, no packets will be dropped. If packets come in at a higher rate, for example 40,000 bytes per second then a certain amount of packets will be dropped.

New in V3.4 is the ability to configure to  $r$  and  $B$ . The user can either take the default configuration type which means that the router will calculate  $r$  and  $B$  based on the configuration or take the custom configuration type and will then be prompted for the token rate in bytes per second and the token bucket size in bytes. As a recommendation the bucket size should be large enough to allow a burst of between five and ten packets. If a custom configuration is used, the bandwidth allocated in the policy dictates how many buffers are occupied whilst the token rate controls at which rate the traffic is policed. You must ensure that the bandwidth allocation is less than or equal to the bandwidth share configured on the interface.

Buffer management, which can drop excess traffic, will be rarely applied to the premium queue as the flow has been policed prior to the packets being placed in the queue. Packets that are sent using the shared buffers will be shown as "ovr snt" when an output queue is monitored from talk 5, feature ds, using the **list streams** command. If packets are dropped by buffer management they will show in the "buf drp" column in the output from the **list streams** command.

Now let's look at the AF/BE queue.



New in V3.4 is the option for the AF/BE queue to be policed. The policing for this queue differs from that of the premium queue. In the premium queue, packets which are out of profile are dropped, but in the AF/BE queue the packets are marked. The two marking approaches available into the IBM routers are single rate three colour marker and two rate single colour marker. These are currently defined in IETF drafts.

Both markers use a dual leaky bucket regulator as shown above. In the single rate marker the two buckets are filled at the same rate whereas in the two rate marker, the buckets are filled at different rates. Both approaches meter the traffic and mark the packet according to whether it exceeds its committed information rate (CIR). A packet is marked as green, yellow or red depending on whether it exceeds its CIR. A green packet has the lowest drop precedence whilst a red packet has the highest drop precedence. The ultimate aim of this approach is to implement some mechanism (such as weighted random error detection or WRED) which will drop the packets, at times of congestion, based on their drop precedence. Until such a mechanism is delivered core routers could be configured with policies which treat the markings differently.

Each token bucket is the combination of a meter and a marker - the meter measures each packet, then passes the packet and the metering result to the marker which marks the packet appropriately.

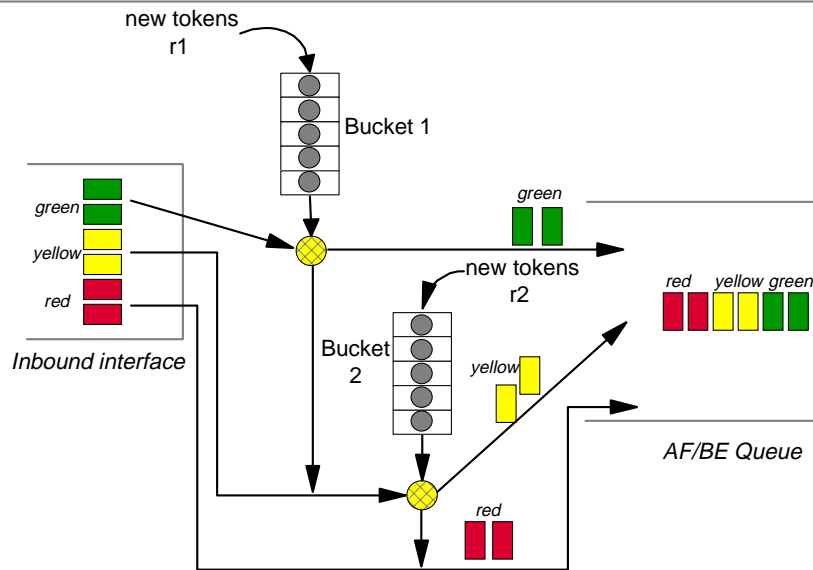
There are two types of meters - those which are colour aware and those which are colour blind. We will look at the colour blind approach first which assumes that all incoming packets are green. If a packet arrives and there are sufficient tokens in bucket 1 the packet will remain green. If there are no tokens or insufficient tokens the traffic will be handled by bucket 2. If there are tokens in bucket 2 to handle the traffic the packets will be marked as yellow. If there are no tokens or insufficient tokens in bucket 2 the packets will be marked as red. If a traffic stream is running at less than its CIR, both buckets will fill. If the traffic has been running at less than its CIR the buckets will fill. Therefore a small burst will be allowed in to the DS domain if there are sufficient numbers of tokens.

The difference between the single and two rate marker is the rate at which the buckets are filled.

1. If a single rate meter is employed, both buckets are filled at an equal rate ( $r1 = r2$ ) which is defined by the CIR. The meter marks the packets according to three parameters - CIR, committed burst size (CBS) and the excess burst size (EBS). The CIR is measured in bytes per second of IP packets (including IP header but not link layer header). CBS and EBS are measured in bytes and are the token bucket depth. The depth of bucket 1 is CBS and the depth of bucket 2 is EBS. It is recommended that CBS and EBS are equal to or

greater than the size of the largest possible IP packet in the stream. Therefore a packet will always be green if its rate is less than CIR and will be green if the excess is less than the number of tokens in bucket 1. If the packet is greater than CIR and greater than the number of tokens in bucket 1, but less than the number of tokens in bucket 2, it will be marked as yellow. If it is greater than the CIR and the number of tokens in both buckets, it will be marked as red.

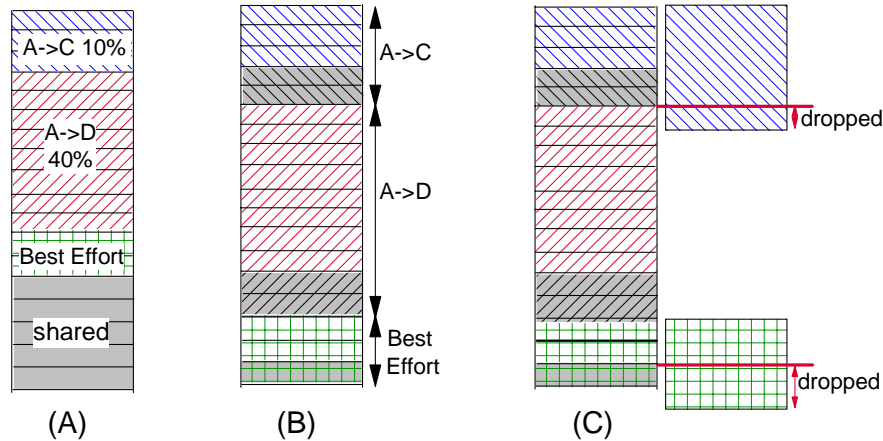
2. In the two rate meter, the buckets are filled at different rates. The first bucket is still filled at the CIR, but bucket 2 is filled at the peak information rate (PIR). The PIR is always greater than the CIR. The depth of bucket 1 is known as the committed burst size (CBS) and the depth of bucket 2 is the peak burst size (PBS). It is recommended that CBS and PBS are equal to or greater than the size of the largest possible IP packet in the stream.



In the colour aware mode incoming packets have already been marked. Packets that were marked cannot be upgraded in the new device - i.e. incoming red packets remain marked as red, and yellow packets will either be marked yellow or re-marked red. The single and two rate markers still act in the same fashion.

## AF / BE Queue

22,000 bytes for QoS allocation &amp; 5,500 bytes of shared buffer



Note the control queue is not shown in the above diagram

Our policy database defined two policies that use the AF/BE queue. The AF/BE queue has 80% of the output bandwidth, of which 80% can be used for QoS allocation. So 64% of the bandwidth is available for QoS allocation with the other 16% being held as a shared pool of resources. The policy also defines for the AF traffic what policing method, if any, is to be employed.

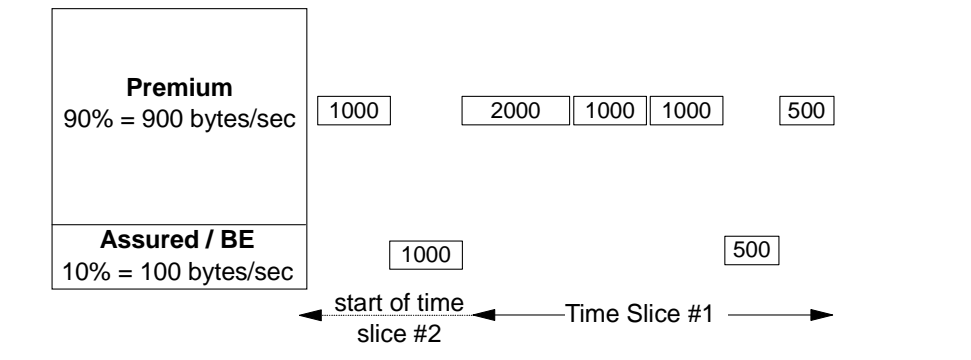
In our DiffServ configuration we have said that best efforts traffic requires a minimum of 10% of the output bandwidth and control requires 5% of the output bandwidth. The best efforts and control traffic are treated as two non-policed streams within the AF/BE queue. Therefore, the sum of all the bandwidth requirements for DS-policies which define the use of the AF/BE queue must be about 50% (80% of the 80 = 64%; less 15% for BE & CTL).

The traffic from A to C will be policed and requires 10% of the output bandwidth whilst traffic from A to D requires 40% of the output bandwidth. The allocation of buffers is not related to the colour marking. Scenario A shows the division of buffers when all the traffic is sending at rate equal to their share of the output bandwidth. Traffic from A to D needs 40% of the output bandwidth, so it will get about 60% (40/64) of the buffers. The AF / BE queue has 27,500 bytes of buffer, 22,000 of which can be used by QoS allocation and the remaining 5,500 buffers are held by the router as shared buffers. Therefore, the traffic from A to D will get 13,750 bytes of buffers in the queue.

Scenario B shows what happens when all the streams exceed their share of the bandwidth. Each stream has taken some of the shared buffers. Remember that each stream can only take out as many buffers as it leaves behind. This actually means that there will always be a small amount of shared buffers that are available (not shown in the diagram). The excess traffic that is placed in the shared buffers will be shown in the "ovr snt" column of the **list streams** command from talk 5, feature ds.

Scenario C shows what happens when two of the streams require more buffers than are available. The excess traffic will be dropped. Traffic that is dropped is shown in the "buf drp" column of the **list streams** command from talk 5, feature ds.

- Scheduler uses Self Clocked Fair Queuing - SCFQ
- Scheduler guarantees each queue a transmission rate
  - ▶ determined from % of bandwidth or rate
  - ▶ in certain time slice :
    - ratio of "amount of traffic forwarded"  $\approx$  ratio of queue weights



The scheduler arbitrates packet transmission across the two queues. Having one queue which is used exclusively for premium traffic ensures that the traffic sees relatively small delays. The scheduling mechanism used is self clocked fair queuing (SCFQ) algorithm, which is a type of weighted fair queuing. When an interface is enabled for DS using the **set interface**, the weights are set to 90% for premium and 10% for AF/BE. This is the default setting. Whilst it should be suitable for most scenarios, it can be tuned if required. The default settings mean that the scheduler checks the queues in this ratio. This is how low delay is guaranteed for traffic assigned to the premium queue. So if you assigned too much traffic to your premium queue this would have an adverse affect on the traffic in the other queue. Recall that the EF traffic was policed to its policy defined rate, so it will not starve the traffic in the other queue.

This is a summary of how SCFQ works. When a packet is forwarded the scheduler calculates how long it took to forward that packet. The equation is  $\text{Time} = \text{service time} + (\text{data size}/\text{queue weight})$ . The scheduler forwards a packet from the queue with smallest service time. Each time a packet is transmitted the service time is incremented. Let's explain this concept using the diagram above. Assume that the rate is 1000 bytes per second. A packet is forwarded from the premium queue and its service time calculated. Traffic will then be forwarded from the AF/BE queue until the premium queue has the smallest service time.

The iterations for first time slice in the diagram above are:

- Calculate time taken forwarding packet from premium queue:  
 $T\text{-prem} = \text{service time} (0) + 500/0.9 = 556$
- As no packets have been forward from AF/BE queue, it must have lowest service time. Send the packet and calculate the time:  
 $T\text{-assur} = \text{service time} (0) + 500/0.1 = 5000$
- Is  $T\text{-assur}$  still less than  $T\text{-prem}$ ? 5000 versus 556, no, forward a packet from the premium queue  
 $T\text{-prem} = 556 + 1000/0.9 = 556 + 1111 = 1667$
- Is  $T\text{-prem}$  still less than  $T\text{-assur}$ ? 1667 versus 5000, yes, forward another packet from premium queue  
 $T\text{-prem} = 1667 + 1000/0.9 = 1667 + 1111 = 2778$
- Is  $T\text{-prem}$  still less than  $T\text{-assur}$ ? 2778 versus 5000, yes, forward another packet from premium queue  
 $T\text{-prem} = 2778 + 2000/0.9 = 2778 + 2222 = 5000$
- Is  $T\text{-prem}$  still less than  $T\text{-assur}$ ? 5000 versus 5000  
 ... etc.

Note that when there are no packets in the premium queue, and there are packets in the AF/BE queue, a packet will be forwarded from the AF/BE queue.

Within the time period the number of bytes forwarded from the premium queue (4500 bytes) is nine times the number of bytes forwarded from the AF/BE queue (500 bytes).

- Each FR circuit has own CIR
    - ✗ cannot treat as one interface
  
  - ✓ Solution : Virtual interface (VIF)
    - ✓ each VIF has an EF and AS/BE queue
    - ▶ scheduling between queues as per PPP
- 

On a frame relay interface there may be multiple logical circuits, each of which is guaranteed its own committed information rate (CIR). Thus the scheduling of queues cannot be independent of the frame relay circuit. A virtual interface (VIF) is used to resolve this problem - a VIF has a one-to-one relationship with a frame relay circuit. The link capacity of the VIF is based on the CIR provided to the circuit. Each VIF has an EF and AS/BE queue and the scheduling is performed as described on the previous page. The VIF is not explicitly configured. When DiffServ is defined on a FR interface, the router creates the appropriate number of VIFs.

- Four Parts

1. Configure speed on WAN interface
2. Define 50 receive buffers on the WAN interface
3. Define DiffServ parameters
4. Define the policies

- Notes :

- ✓ RSVP & DiffServ can be enabled on the same interface
  - ✗ Cannot have DiffServ and BRS on the same interface
- 

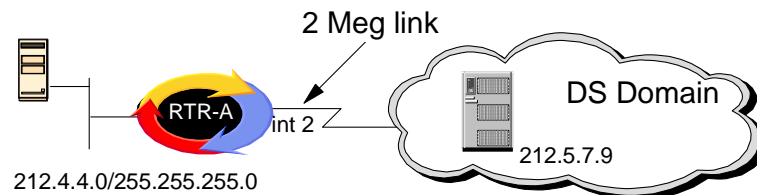
As we have reviewed the background to DiffServ and looked at the implementation in IBM routers, the next section will focus on how to configure DiffServ. There are four main parts:

1. Configuring the speed or CIR on the WAN interface - the speed must be configured in advance of configuring DS. If it is not configured you will be informed of this when you enable DS on an interface.
2. Define 50 receive buffers on the WAN interface
3. Define the DiffServ parameters - enable DS globally and enable DS on PPP or FR interfaces of choice. In the DS feature you can tune the packet size and the buffer details.
4. Define the policies - define which traffic should be handled by DS and how DS traffic should be marked.

Note that from FR, DiffServ and BRS cannot be enabled on the same interface so all PVCs will either be prioritised on the basis of DiffServ or BRS.

- Sample network: requirements

- ✓ All traffic to 212.5.7.9 to get 19% of the total output bandwidth, and experience minimal delays
- ✓ All other traffic to 212.5.7.0 LAN to get 10% of the total output bandwidth



In the coming pages the above requirements will be configured. All traffic going to 212.5.7.9 is to get 19% of the total output whereas traffic going to any other device on the 212.5.7.0 LAN will get 10% of the output bandwidth. The traffic going to 212.5.7.9 requires minimal delays and will therefore be handled by the premium queue.

This is a basic example to show you the format of the commands. In a network you would need to define policies for traffic flowing in both directions to get a sensible QoS definition.

This configuration uses PPP links - the configuration of DS and the policy would be identical for frame relay.

## ■ 1. Configure Diff Serv

```
rtr-a *TALK 6
Gateway user configuration
rtr-a Config>FEATURE DS
Differential Services Config
rtr-a DS Config>ENABLE DS
DiffServ enabled
rtr-a DS Config>SET INTERFACE
Enter Diffserv Interface number [0]? 2
Set Premium Queue Bandwidth (%) (1 - 99) [20]?
    Assured Queue Bandwidth (%) = 80
Configure Advanced setting (y/n)? [No]:
Accept input (y/n)? [Yes]:
rtr-a DS Config>EXIT
rtr-a Config>
```

## ■ 2. Add the policies

```
rrtr-a Config>FEATURE Policy
IP Network Policy configuration
rtr-a Policy config>
```

I have assumed that the interface and IP configuration is complete. For DS the speed must be configured on the PPP interface, or DS will assume a link speed on 1 Mbps. Also the receive buffers should be increased to (at least) 50 receive buffers on the WAN interface (interface 2).

DiffServ is configured in two parts - first it is enabled globally and then it is enabled on each interface. The advanced configuration allows you to tune the scheduler, QoS allocation and the buffer size. For most configuration you will not need to modify the advanced settings. If you are experiencing problems, such as a lack of buffers or you wish to decrease the delay experienced by traffic in the AF/BE queue you can tune the advanced configuration settings. The default advanced settings are:

- 90% weight for premium queue - this is used by the SCFQ scheduler. So for example if you wanted to decrease the delay experienced by traffic in the AF/BE queue, you would decrease this value
- 5500 bytes for the egress buffer size for the premium queue and 27500 bytes for the egress buffer size for the AF/BE queue. If you found that your traffic demanded more buffers, you could increase this value. Do this with extreme care and always confirm that you have enough memory before changing this parameter. Available memory can be determined from talk 5 using the **memory** command and looking at the Never alloc column. As a very approximate guide for tuning, the queue sizes should be in a ratio of 1 to 5.
- 95% maximum allocation for premium queue, so only 95% of the 5500 bytes assigned to the premium queue can be assigned to user's data. The extra 5% are held as shared buffers. This value also means that the sum of the bandwidth requirements, defined in the DS-actions, for traffic assigned to the premium queue cannot exceed 19% of the output bandwidth (95% of 20%).
- 80% maximum allocation for the AF/BE queue, so only 80% of 27500 buffers assigned to the premium queue can be assigned to user's data. The extra 20% are held as shared buffers. This value also means that the sum of the bandwidth requirements, for traffic handled by the AF/BE queue cannot exceed 64% of the output bandwidth (80% of 80%).

Also in DS configuration you can tune the minimum percentages to be allocated for best efforts and control traffic. Recall the both the streams are handled by the AF/BE queue and these streams are never policed. The commands to do this are **set be-alloc-min** and **set ctl-alloc-min**.

The next step is to define the policy.

```
rtr-a Policy config>ADD POLICY
Enter a Name (1-29 characters) for this Policy []? ef-to-212.5.7.9
Enter the priority of this policy (This number is used to
determine the policy to enforce in the event of policy conflicts) [5]? 10
List of Profiles:
    0: New Profile
Enter number of the profile for this policy [0]?
Profile Configuration questions. Note for Security Policies, the Source
Address and Port Configuration parameters refer to the Local Client Proxy
and the Destination Address and Port Configuration parameters refer to the
Remote Client Proxy
Enter a Name (1-29 characters) for this Profile []? to-212.5.7.9
Source Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]?
Enter IPV4 Source Address [0.0.0.0]? 212.4.4.0
Enter IPV4 Source Mask [255.255.255.0]?
Destination Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]? 3
Enter IPV4 Destination Address [0.0.0.0]? 212.5.7.9
Protocol IDs:
    1) TCP
    2) UDP
    3) All Protocols
    4) Specify Range
Select the protocol to filter on (1-4) [3]?
```

Policies can be defined with two different starting points - either using the **add policy** command which will prompt you through creating any details / actions that are not defined, or defining the actions using individual commands. I have found it quicker to define the policy using the **add policy** command, so this approach is illustrated on the next few pages.

The first detail you are prompted for is the name of the policy and the priority of the policy. The larger the number the higher the priority of the policy. When the traffic is passed through the policy database, it is compared to the highest priority profile first, and the first matching action will be taken. In the network we want traffic going a specific address, 212.5.7.9, to get a different handling from devices on the same subnet. Therefore, the profile going to 212.5.7.9 must have a higher priority than that assigned to the 212.5.7.0 network. So this policy has been given a priority of 10, and the other policy will have a priority of 5.

As a traffic profile does not exist the router prompts for the creation of one. A name is requested for this profile, try to keep the names as meaningful as possible as a traffic profile can be used by another policy. The traffic profile will be called "to-212.5.7.9". The router can inspect the source and destination IP addresses, the IP protocol, source and destination ports and the DS byte (i.e. DS codepoint) to determine if the traffic matches a policy. At this time we are defining a profile that says all traffic from the 212.4.4.0 network going to 212.5.7.9 will match this policy. Both the source and the destination addresses can be defined as a netmask, range or single address. The format of the netmask and single address are shown above. If you specify range you will be asked for a starting address and a finishing address.

The router then asks if you wish this policy to apply to TCP traffic only, UDP traffic only, all protocols or a specific range. We will take all protocols in this example. If you take option 4 you will be asked for a start and finish protocol value.



## Sample Configuration - 3



```
Enter the Starting value for the Source Port [0]?
Enter the Ending value for the Source Port [65535]?
Enter the Starting value for the Destination Port [0]?
Enter the Ending value for the Destination Port [65535]?
Enter the Mask to be applied to the Received DS-byte [0]?
Enter the value to match against after the Mask has
been applied to the Received DS-byte [0]?
Configure local and remote ID's for ISAKMP? [No]:
Limit this profile to specific interface(s)? [No]:
Here is the Profile you specified...
Profile Name      = to-212.5.7.9
  sAddr:Mask=    212.4.4.0 : 255.255.255.0   sPort=    0 : 65535
  dAddr      =    212.5.7.9 :                dPort=    0 : 65535
  proto      =                0 : 255
  TOS        =                x00 : x00
  Remote Grp=All Users
Is this correct? [Yes]:
List of Profiles:
  0: New Profile
  1: to-212.5.7.9
Enter number of the profile for this policy [1]? 1
```

This profile defines that the action will be applied to all source and destination port numbers and any setting of the TOS/DS-byte.

*Notes on mask and value:* A mask of 1 indicates that we want to look at that bit, whereas a mask of 0 implies that we are not concerned with the setting of that bit. For example, a policy that should only act on packets with the fifth bit set. The mask would be 00001000 = 0x08 and the value would be 0x08. A mask of 0x08 implies that we are only concerned with the setting of the fifth bit so the DS byte could be 11101000, 10001000, etc. If the mask had been defined as 0xFF with a value of 0x08, the policy would only apply to traffic with a DS byte of 00001000.

IKE is not applied in this policy so IDs are not configured. Policies can also be restricted to traffic which is received over interface x and leaving over interface y (interface being IP interface). With a traffic profile defined, we are now asked again which profile should this policy use. Selecting option 1 takes the profile that has just been created. If you wished to create further profiles, you could do so by taking option 0 and being guided through another profile configuration.



## Sample Configuration - 4



```
List of Validity Periods:
  0: New Validity Period
  1: allTheTime (Cannot DELETE or CHANGE)
  2: allTheTimeMonThruFri (Cannot DELETE or CHANGE)
  3: 9to5MonThruFri (Cannot DELETE or CHANGE)
  4: 5to9MonThruFri (Cannot DELETE or CHANGE)
Enter number of the validity period for this policy [1]? 1
Do you wish to Map a DiffServ Action to this Policy? [No]: y
DiffServ Actions:
  0: New DiffServ Action
  1: EF
  2: AF11
  3: AF21
  4: AF31
  5: AF41
Enter the Number of the DiffServ Action [1]? 1
Do you wish to Map a RSVP Action to this Policy? [No]:
Policy Enabled/Disabled (1. Enabled, 2. Disabled) [1]?
Here is the Policy you specified...
```

The next section that the policy requires is "when is this policy to be active". The router provides four time templates. If you do not like any of the templates you can take option 0 and the router will prompt you through the creation of a validity period. An example of creating a validity period is shown below. The validity period is going to define a policy which is only valid on Mondays between 11:00 and 12:00.

```
rtr-a Policy config>ADD VALIDITY-PERIOD
Enter a Name (1-29 characters) for this Policy Valid Profile []?
mon-11-12
Enter the lifetime of this policy. Please input the information in
the following format:
          yyyyymmddhhmmss:yyyyymmddhhmmss OR '*' denotes
forever.
[*]?
During which months should policies containing this profile be
valid. Please input any sequence of months by typing in the first
three letters of each month with a space in between each entry, or
type ALL to signify year round.
[ALL]?
During which days should policies containing this profile be valid.
Please input any sequence of days by typing in the first three
letters of each day with a space in between each entry, or type ALL
to signify all week
[ALL]? mon
Enter the starting time (hh:mm:ss or * denotes all day)
[*]? 11:00:00
Enter the ending time (hh:mm:ss)
[00:00:00]? 12:00:00
Here is the Policy Validity Profile you specified...
Validity Name      = mon-11-12
Duration          = Forever
Months            = ALL
Days              = MON
Hours             = 11:00:00 : 12:00:00
Is this correct? [Yes]:
rtr-a Policy config>
```

Returning to our network we want the policy to be valid all the time so we select option 1 for all the time. The next step is to define our DiffServ action. (Note that if you have an encryption image you will be asked if you wish to define an ipsec-action before being asked about a DiffServ action).

The router provides five templates for DiffServ actions. They are:

```
rtr-a Policy config>LIST DIFFSERV-ACTION ALL

DiffServ Name      = EF                                     Type =Permit
  DS mask:modify   =xFC:xB8
  Queue:BwShare    =Premium      : 19 %
  Token Rate:      = 0 bytes/sec
  Token Bucket:    = 0 bytes

DiffServ Name      = AF11                                    Type =Permit
  DS mask:modify   =xFC:x28
  Queue:BwShare    =Assured/BE   : 15 %
  No Policing Selected

DiffServ Name      = AF21                                    Type =Permit
  DS mask:modify   =xFC:x48
  Queue:BwShare    =Assured/BE   : 10 %
  No Policing Selected

DiffServ Name      = AF31                                    Type =Permit
  DS mask:modify   =xFC:x68
  Queue:BwShare    =Assured/BE   : 10 %
  No Policing Selected

DiffServ Name      = AF41                                    Type =Permit
  DS mask:modify   =xFC:x88
  Queue:BwShare    =Assured/BE   : 5 %
  No Policing Selected
rtr-a Policy config>
```

We will have a look at creating a DiffServ action when we create the next policy. A DiffServ actions tells the router that when the traffic matches the conditions of this policy which queue the traffic will use, what its bandwidth requirement is and how the packet is to be marked. The marking is defined in terms of a mask and a value. The mask indicates which bits to be changed and the value states what they will be changed to. A one in the mask means that the bit will be changed and a zero, the bit will be unchanged. A one in the value implies that that bit will be set. So a mask of 0xFC states that the first six bits of the TOS byte / DS codepoint will be changed. 0xFC is the most common mask as this means that the DSCP is set (recall that the DSCP is the first 6 bits of the old TOS byte and the last two bits are currently undefined).

If the incoming TOS byte is 00000000, a value of 0xB8 means that codepoint will be marked as 10111000, 0x28 is a marking of 00101000, 0x48 010010000, 0x68 01100000 and 0x88 100010000. If the incoming TOS bytes as 01000011, the mask was 0xFC and the value was 0xB8 the packet would be marked as 10111011.

The EF template can be used for the traffic going to 212.5.7.9 so option 1 is selected. The token rate and token bucket of "0 bytes/sec" means that the actual values will be determined based on the line speed. We are then asked if we want to define an RSVP action - no . The router then lists the configuration ....



## Sample Configuration - 5



```
Policy Name      = ef-to-212.5.7.9
  State:Priority =Enabled      : 5
  Profile        =to-212.5.7.9
  Valid Period   =allTheTime
  DiffServ Action=EF

Is this correct? [Yes]:
You must enable and configure DiffServ in feature DS before
QOS can be ensured for this policy
rtr-a Policy config>ADD POLICY
Enter a Name (1-29 characters) for this Policy []? to-212.5.7.0
Enter the priority of this policy (This number is used to
determine the policy to enforce in the event of policy conflicts) [5]?
List of Profiles:
  0: New Profile
  1: to-212.5.7.9
Enter number of the profile for this policy [1]? 0
Profile Configuration questions. Note for Security Policies, the Source
Address and Port Configuration parameters refer to the Local Client Proxy
and the Destination Address and Port Configuration parameters refer to the
Remote Client Proxy
Enter a Name (1-29 characters) for this Profile []? to-212.5.7.0
Source Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]?
```

If the information is incorrect you can answer no the “Is this correct?” question and the router will then guide you through the policy configuration again.

This same policy could be have been created by defining a traffic profile called “to-212.5.7.9” using the **add profile** command and then using **add policy** command to “join” all the items together.

The next policy to create is that for all traffic going to the 212.5.7.0 network. It is important that this policy is of lower priority than that going to 212.5.7.9 or the wrong policy will be applied. The policy going to the network has been assigned a value of 5 - lower than that for one going to 212.5.7.9.

We need to create a new traffic profile for this policy and the traffic profile will be called “to-212.5.7.0”.



## Sample Configuration - 6



```
Enter IPv4 Source Address [0.0.0.0]? 212.4.4.0
Enter IPv4 Source Mask [255.255.255.0]?
Destination Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]?
Enter IPv4 Destination Address [0.0.0.0]? 212.5.7.0
Enter IPv4 Destination Mask [255.255.255.0]?
Protocol IDs:
  1) TCP
  2) UDP
  3) All Protocols
  4) Specify Range
Select the protocol to filter on (1-4) [3]?
Enter the Starting value for the Source Port [0]?
Enter the Ending value for the Source Port [65535]?
Enter the Starting value for the Destination Port [0]?
Enter the Ending value for the Destination Port [65535]?
Enter the Mask to be applied to the Received DS-byte [0]?
Enter the value to match against after the Mask has
been applied to the Received DS-byte [0]?
Configure local and remote ID's for ISAKMP? [No]:
Limit this profile to specific interface(s)? [No]:
```

---

The new profile is very similar to that of "to-212.5.7.9" except that the destination address is a Netmask, with a destination address of 212.5.7.0 and a class C mask.



## Sample Configuration - 7



```
Here is the Profile you specified...
Profile Name      = to-212.5.7.0
sAddr:Mask=      212.4.4.0 : 255.255.255.0   sPort=    0 : 65535
dAddr      =      212.5.7.0 : 255.255.255.0   dPort=    0 : 65535
proto       =              0 : 255
TOS         =              x00 : x00
Remote Grp=All Users

Is this correct? [Yes]:
List of Profiles:
  0: New Profile
  1: to-212.5.7.9
  2: to-212.5.7.0

Enter number of the profile for this policy [1]? 2
List of Validity Periods:
  0: New Validity Period
  1: allTheTime (Cannot DELETE or CHANGE)
  2: allTheTimeMonThruFri (Cannot DELETE or CHANGE)
  3: 9to5MonThruFri (Cannot DELETE or CHANGE)
  4: 5to9MonThruFri (Cannot DELETE or CHANGE)

Enter number of the validity period for this policy [1]? 1
Do you wish to Map a DiffServ Action to this Policy? [No]: y
```

Once the profile has been created, the router lists it and then offers a list of the available profiles. We will take option 2 which is one we have just created. We are going to use the “allTheTime” validity period for this policy. The next step is to define the DiffServ action.



## Sample Configuration - 8



```
Do you wish to Map a DiffServ Action to this Policy? [No]: yes
DiffServ Actions:
  0: New DiffServ Action
  1: EF
  2: AF11
  3: AF21
  4: AF31
  5: AF41
Enter the Number of the DiffServ Action [1]? 0
Enter a Name (1-29 characters) for this DiffServ Action []? af1
Enter the permission level for packets matching this DiffServ
Action (1. Permit, 2. Deny) [2]? 1
List of Forwarding Services:
  1) Premium
  2) Assured/BE
Enter the Forwarding Service(1-2) for this DiffServ Action [2]?
How do you want to specify the bandwidth allocated to this service?
Enter absolute kbps(1) or percentage of output bandwidth(2) [2]?
Enter the percentage of output bandwidth allocated to this service [10]?
```

---

For the traffic going to 212.5.7.0 network we are going to use the AF queue and perform policing via a single rate, colour blind marker. I have chosen to use a colour blind marker because this router is at the edge of the DS domain and therefore the received packets will not be marked. Selecting option 0 guides us through the creation of a new DS action, which we will call "af1". We want our packets to be forwarded by the AF/BE queue and we want it to have 10% of the output bandwidth.



## Sample Configuration - 9



```
List of Assured Forwarding Class:
 1) AF1 Class DS Byte
 2) AF2 Class DS Byte
 3) AF3 Class DS Byte
 4) AF4 Class DS Byte
 5) New Class DS Byte
Enter the AF Class (1-5) for outgoing packets matching
this DiffServ Action [5]? 1
List of Policing Type in AF Class:
 1) Single Rate Color Blind TCM
 2) Single Rate Color Aware TCM
 3) Two Rate Color Blind TCM
 4) Two Rate Color Aware TCM
 5) None
Enter the AF Class(1-5) Policing for outgoing packets matching
this DiffServ Action [5]? 1
Single Rate TCM:
Committed Info Rate (CIR in bytes/sec) [0]?
Committed Burst Size (CBS in bytes) [4000]?
Excess Burst Size (EBS in bytes) [4000]?

Here is the DiffServ Action you specified...
```

The router then asks you how the packet should be marked. You are offered five choices. All the classes have a mask of 0xFC. The AF1 class DS byte has a value 0x20, AF2 has a value of 0x40, AF3 0x60 and AF4 has a value of 0x80. Option 5 lets you define your own mask and value. We are going to select the AF1 class which means that packets matching this policy will have the first 6 bits (mask 0xFC = 11111100) of their TOS-byte changed to 0x20 i.e. 001000 plus whatever that last two bits were already. The next question relates to the type of policing. We will use a single rate colour blind marker - option 1. This means that the packets will be following DSCP:

1. If the traffic is in profile, i.e. green, its DSCP will be 001010.
2. If the traffic is marked as yellow, its DSCP will be 001100.
3. If the traffic is marked as red, its DSCP will be 001110.

We are asked for the CIR, CBS and EBS. A CIR of 0 means that the router will calculate the bytes per second based on the line speed and the percentage of the output bandwidth requested. In this example, our line speed is 2 Mbps and this action requires 10% of that. That is 256,000 bytes per second. Both token buckets are filled at 256,000 bytes and have a depth of 4,000 bytes.

Option 2, the colour aware marker, asks for the same parameters. If you select either option 3 or 4, the questions are:

```
Two Rate TCM:
Committed Info Rate (CIR in bytes/sec) [0]?
Committed Burst Size (CBS in bytes) [4000]?
Peak Info Rate (PIR in bytes/sec) [0]?
Peak Burst Size (PBS in bytes) [4000]?
```

With the DiffServ action defined, the router lists the details ...



## Sample Configuration - 10

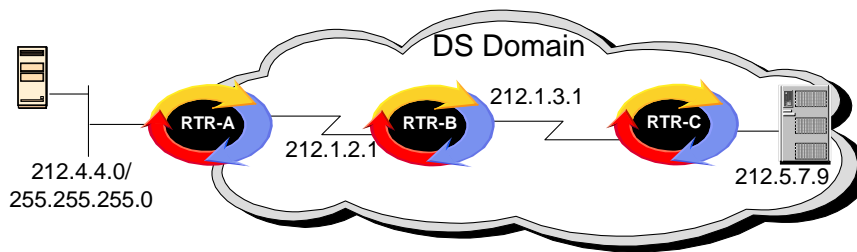


```
DiffServ Name   = af1                               Type =Permit
DS mask:modify  =xFC:x20
Queue:BwShare   =Assured/BE : 10 %
TCM:Class = SR,CB:AF1
CIR = 0 bytes/sec; CBS = 4000 bytes
EBS = 4000 bytes
Is this correct? [Yes]:
DiffServ Actions:
0: New DiffServ Action
1: EF
2: AF11
3: AF21
4: AF31
5: AF41
6: af1

Enter the Number of the DiffServ Action [1]? 6
Do you wish to Map a RSVP Action to this Policy? [No]:
Policy Enabled/Disabled (1. Enabled, 2. Disabled) [1]?
Here is the Policy you specified...

... etc (not shown to save space)
```

We then select option 6 which takes our “af1” action. This completes the configuration of the DS-boundary router. Now lets consider the configuration of the interior DS routers.



Two policies:

1. all traffic marked B8 to be handled by EF queue
2. all traffic marked FC/20 to be handled by AF queue

So how does the policy differ for the interior DS routers?

1. The DS standard defines that the traffic profile on the interior nodes need only inspect the DSCP - but if you wished to look at other parts of the IP header there is nothing to stop you. So the traffic going to 212.5.7.9 will have been marked as 0xB8 by RTR-A, so RTR-B needs a policy that states that traffic with this marking should be handled by the premium queue and receive 19% of the output bandwidth. The traffic going to 212.5.7.0 will have been marked by RTR-A with 001xxx where xxx is dependent on the drop precedence so RTR-B will need to have a policy which places traffic with a 001xxx DSCP in the AF queue.
2. The validity period would be the same as for the boundary nodes.
3. The DS-action would need to provide a consistent QoS allocation. For the EF policy, the DS-action should not change the DSCP so the mask and value in the action would be zero. For the AF policy, a single rate colour aware marker should be chosen as packet will have already been marked as red, green or yellow by RTR-A. The DSCP is not modified by the policy - the first three bits must remain unchanged, and the second three bits will be updated by the marker. For both the EF and AF policies you will need to create new DS actions as the all the provided templates remark the DSCP.

As the procedure for configuring RTR-B is very similar to that of RTR-A, only two pieces will be reviewed:

1. Defining the traffic profile for traffic destined for 212.5.7.0 network. Recall that in RTR-A these packets were handled by AF1 with a single rate colour blind marker. That means the packets will all start with 001.
2. Defining the DiffServ action for the AF1 traffic.

### ■ Profile configuration for AF1 policy

```
rtr-b Policy config>ADD PROFILE
Profile Configuration questions. Note for Security Policies, the Source
Address and Port Configuration parameters refer to the Local Client Proxy
and the Destination Address and Port Configuration parameters refer to the
Remote Client Proxy
Enter a Name (1-29 characters) for this Profile []? 20
Source Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]?
Enter IPV4 Source Address [0.0.0.0]?
Enter IPV4 Source Mask [0.0.0.0]?
Destination Address Format (1:NetMask, 2:Range, 3:Single Addr) [1]?
Enter IPV4 Destination Address [0.0.0.0]?
Enter IPV4 Destination Mask [0.0.0.0]?
Protocol IDs:
  1) TCP
  2) UDP
  3) All Protocols
  4) Specify Range
Select the protocol to filter on (1-4) [3]?
```

Interior DS nodes need only look at the DS codepoint to decide how to handle traffic. This means that the router does not need to inspect the source or destination address. Therefore the profile is wild-carded for the addresses. We also don't care about the protocol IDs ...



## Interior DS Node Configuration - 2



```
Enter the Starting value for the Source Port [0]?
Enter the Ending value for the Source Port [65535]?
Enter the Starting value for the Destination Port [0]?
Enter the Ending value for the Destination Port [65535]?
Enter the Mask to be applied to the Received DS-byte [0]? e0
Enter the value to match against after the Mask has
been applied to the Received DS-byte [0]? 20
Configure local and remote ID's for ISAKMP? [No]:
The Source and/or Destination Address information you specified
includes all addresses. You should specify an Interface Pair
with this profile to further qualify what traffic you wish to filter
to this policy.
Limit this profile to specific interface(s)? [No]: yes
Interface Pair Groups:
    0: New Ifc Pair
Number of Ifc Pair Group [1]? 0
Enter a Group Name (1-29 characters) for this Interface Pair []? ac
Ingress Interface IP Address (255.255.255.255 = any ingress)
[255.255.255.255]? 212.1.2.1
Egress Interface IP Address (255.255.255.255 = any egress)
[255.255.255.255]? 212.1.3.1
```

... the source or destination port. But we do care about the DS-byte. In RTR-A we chose to use the AF1 DSCP. The first three bits of this codepoint are 001 and the last three bits are set depending on whether the packet is marked as green, yellow or red. So therefore our mask should be  $0xE0 = 1110000$  and the match  $0x20 = 00100000$ . If the AF2 DSCP had been chosen, the match would have been  $0x40$ ,  $0x60$  for AF3 and  $0x80$  for AF4.

As we have not defined any source or destination addresses the router asks that an interface pair group should be defined. Note that you do not have to define the pair for DiffServ as the application of the policy does not affect the route taken by the packet. With an interface pair, the policy will only be applied when a match occurs on the required fields in the IP header and if the packet is received on the a particular interface and leaves via a particular interface. In our network we know that packets destined to the 212.5.7.0 network will arrive on 212.1.2.1 and leave via the 212.1.3.1 interface so these are our ingress and egress interfaces.



## Interior DS Node Configuration - 3



```
Interface Pair Groups:
  0: New Ifc Pair
    1) Group Name: ac
        In:Out=      212.1.2.1 : 212.1.3.1
Number of Ifc Pair Group [1]?
Here is the Profile you specified...
Profile Name      = 20
sAddr:Mask=      0.0.0.0 : 0.0.0.0      sPort=      0 : 65535
dAddr:Mask=      0.0.0.0 : 0.0.0.0      dPort=      0 : 65535
proto           =          0 : 255
TOS             =          xE0 : x20
Remote Grp=All Users
1. In:Out=      212.1.2.1 : 212.1.3.1
Is this correct? [Yes]:
rtr-b Policy config>
```

The rest of the output of the command is shown above.

■ Diff Serv action for policy AF1:

```
rtr-b Policy config>ADD DIFFSERV-ACTION
Enter a Name (1-29 characters) for this DiffServ Action []? AF1
Enter the permission level for packets matching this DiffServ
Action (1. Permit, 2. Deny) [2]? 1
List of Forwarding Services:
  1) Premium
  2) Assured/BE
Enter the Forwarding Service(1-2) for this DiffServ Action [2]?
How do you want to specify the bandwidth allocated to this service?
Enter absolute kbps(1) or percentage of output bandwidth(2) [2]?
Enter the percentage of output bandwidth allocated to this service [10]?
List of Assured Forwarding Class:
  1) AF1 Class DS Byte
  2) AF2 Class DS Byte
  3) AF3 Class DS Byte
  4) AF4 Class DS Byte
  5) New Class DS Byte
Enter the AF Class (1-5) for outgoing packets matching
this DiffServ Action [5]?
Transmitted DS-byte mask [0]?
Transmitted DS-byte modify value [0]?
```

The next two screens show the creation of the DiffServ action which will put packets which match the traffic profile into the AF/BE queue and use a single rate colour aware marker. This DS action should not change the class of the DS byte. Recall that the DiffServ standard defines the packets are marked as they enter a DS domain and are then not changed. RTR-A put the packets in AF1 class which means that the first three bits are 001 and the second three bits depend on the rate at which the stream is flowing. So the policy in RTR-B should not change the DS-byte. Therefore we need to create a “New Class DS byte” which has a transmitted mask of 0 and a modify value of 0. The second three bits, which reflect the colour marking, will still be updated because they are controlled by the meter and not by the policy.



## Interior DS Node Configuration - 5



```
List of Policing Type in AF Class:
  1) Single Rate Color Blind TCM
  2) Single Rate Color Aware TCM
  3) Two Rate Color Blind TCM
  4) Two Rate Color Aware TCM
  5) None
Enter the AF Class(1-5) Policing for outgoing packets matching
this DiffServ Action [5]? 2
Single Rate TCM:
Committed Info Rate (CIR in bytes/sec) [0]?
Committed Burst Size (CBS in bytes) [4000]?
Excess Burst Size (EBS in bytes) [4000]?
Here is the DiffServ Action you specified...
DiffServ Name   = AF1                               Type =Permit
  DS mask:modify =x00:x00
  Queue:BwShare  =Assured/BE : 10 %
  TCM:Class = SR,CA:CUST
  CIR = 0 bytes/sec;   CBS = 4000 bytes
  EBS = 4000 bytes
Is this correct? [Yes]:
rtr-b Policy config>
```

I have only shown the configuration of the traffic profile - clearly the rest of the policy would need to be defined for traffic going to 212.5.7.0 and also a policy created for traffic marked with 0xB8.

- What can be non-disruptively reconfigured?
  - ✓ details in feature policy - everything
    - ✓ reset from talk 5
  - ✗ details in feature DS - nothing
    - ✗ restart / reload

If DiffServ has been configured and details are changed, some changes can be brought online without needing a whole router restart/reload. If the policy is changed, the database can be rebuilt. This is achieved by:

```
* talk 5
+feature policy
IP Network Policy console
Policy console>reset database
Policy Database reset successful
Policy console>
```

The policy database is always rebuilt when the router is restarted/reloaded. It can also be reset using SNMP set commands.

No changes from feature ds can be activated without a restart/reload.

- 1. From feature policy

- ▶ a) confirm that rules have been generated and used

```
rtr-a *TALK 5
rtr-a +FEATURE POLICY
IP Network Policy console
rtr-a Policy console>LIST STATS
+-----+
|Name                                     |Hits  |
+-----+
|to-212.5.7.9                             |    3|
|   EF                                     (DS) |    3|
+-----+
```

```
rtr-a Policy console>
```

Up to now we have looked at how to configure DiffServ. We will now look at how you can monitor DS and check the configuration. There are three different places to look at DiffServ - the policy, the DiffServ configuration and ELS messages. Lets start with the policy.

The router is configured with a policy which defines how you wish the network to operate. The router translates the policy information into a set of rules that it compares to traffic flows. First, confirm that the rules have been generated. There are many commands available - see the features guide for more details, but the **list stats** command shows the policies and if they have been hit - i.e. has traffic been received which has matched the policy.

## ► b) confirm that correct policy is used

```
rtr-a Policy console>TEST FORWARDER-QUERY
Enter IPV4 Source Address [0.0.0.0]? 212.4.4.1
Enter IPV4 Destination Address [0.0.0.0]? 212.5.7.9
Enter the value for the Source Port [0]?
Enter the value for the Destination Port [0]?
Enter the value for the IP Protocol ID [0]?
Enter the value for the IP TOS Byte [0]?
Results of Test:
Policy (DS):          ef-to-212.5.7.9
DS Action:           EF
rtr-a Policy console>TEST FORWARDER-QUERY
Enter IPV4 Source Address [0.0.0.0]? 212.4.4.1
Enter IPV4 Destination Address [0.0.0.0]? 212.5.7.10
Enter the value for the Source Port [0]?
Enter the value for the Destination Port [0]?
Enter the value for the IP Protocol ID [0]?
Enter the value for the IP TOS Byte [0]?
Results of Test:
Policy (DS):          to-212.5.7.0
DS Action:           af1
rtr-a Policy console>
```

If you want to confirm that the policy is working as you intended you can use the **test forwarder-query** command and enter the IP details.

- 2. Using ELS messages
  - ▶ two subsystems : PLCY & DS

```
17:40:29 PLCY.023: Query(IP), src:212.1.2.2/0,dst:212.5.7.9/0,prot:1,DS:B8
17:40:29 PLCY.024: Result: Rule:b8, Action:ef, Handle:31CC4DC0
17:40:29 DS.024: dsact: QPri=1, RateT=2, Rate=19
17:40:29 DS.039: ds_api: Sid: 0 EF Policer: rate: 48640 peak: 4000 token: 5225
17:40:29 DS.030: dsapi: in 2, out 1. Added Stream 25
17:40:29 PLCY.023: Query(IP), src:212.5.7.9/0,dst:9.180.180.173/0,prot:1,DS:0
17:40:29 PLCY.024: Result: Rule:None, Action:None, Handle:0

17:48:58 PLCY.023: Query(IP), src:212.1.2.2/0,dst:212.5.7.8/0,prot:1,DS:28
17:48:58 PLCY.024: Result: Rule:20-af, Action:af, Handle:31CC5070
17:48:58 DS.024: dsact: QPri=2, RateT=2, Rate=10
17:48:58 DS.030: dsapi: in 2, out 1. Added Stream 26
17:48:58 PLCY.023: Query(IP), src:212.5.7.8/0,dst:212.1.2.2/0,prot:1,DS:0
17:48:58 PLCY.024: Result: Rule:None, Action:None, Handle:0
```

ELS can also be used to monitor DiffServ. There are two subsystems that are particularly useful - policy (**display subsystem plcy all**) and DiffServ (**display subsystem ds all**). The IP subsystem also has relevant messages - in particular IP.124 is useful for showing the incoming TOS setting..

*Note that the policies used for these screen captures are slightly different from those discussed previously. The policy on RTR-A had defined the source address as any, i.e. address and mask 0.0.0.0 rather than 212.4.4.0 network). Also the names of the policies are slightly different.*

These ELS traces were taken on RTR-B, the interior DS node. The first set of ELS messages shows that a packet going from 212.1.2.2 to 212.5.7.9 which has a DSCP of 0xB8 matched policy b8. You can see that that policy defines the use of the premium queue as shown by QPri=1. The rate is 48,640 bytes per second which is 19% of the output bandwidth ( $0.19 \times 2048000 \div 8$ ). You can see that stream 25 has been created and the stream is coming in on interface two and leaving on interface one.

The second set of ELS messages shows that a packet going from 212.1.2.2 to 212.5.7.8 which has a DSCP of 0x28 matched policy 20-af. You can see that that policy defines the use of the AF queue as shown by QPri=2.

- 3. From feature DS
  - ▶ a) determine enabled interface

```
rtr-b DS Console>LIST INTERFACE
DiffServ interfaces:
Net Status   Kbits/s   VirtTime  InMax   InCurr  InShar  InMaxA  InCurA  NumI  NumO
-----
0 Disabled    0         n/a      550000  0       550000  550000  0       4    n/a
1 Enabled    2048      0        27500  0       27500   22000   0       4     4
2 Enabled    2048      0        27500  0       27500   22000   0       6     2
3 Disabled    0         n/a      550000  0       550000  550000  0       4    n/a
4 Disabled   16000     n/a      27500  0       27500   22000   0       4    n/a
rtr-b DS Console>
```

From feature DS, you can confirm which interfaces are DS enabled. The InMax column is the number of bytes of buffers for the interface. The InCurr is the number of bytes currently in use, which was zero at the time of the screen capture. The InMaxA is the maximum allocated number of buffers. This is 80% of InMax as the DS configuration stated that 80% of the AF/BE buffer was available for allocation. InCurA is the amount of buffers currently in use. NumI is the number of incoming streams and NumO is the number of outgoing streams. Remember that by default there are always two streams - BE and CTL.

## ► b) view DS cache

```

rtr-b DS Console>DSCACHE ACTIONS
Source Address to list []?
Destination Address to list []?
Source      Destination      Pro ProtocolInf Net TosIn/Out Action StrmID
212.5.7.9   212.4.4.1        1 T:x08 C:x00    1 x00->x00 PASS none
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x30->x30 PASS 26
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x38->x38 PASS 26
212.4.4.1   212.5.7.9        1 T:x00 C:x00    2 xB8->xB8 PASS 25
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x28->x28 PASS 26
212.5.7.9   9.180.180.66     17 138> 138 1 x00->x00 PASS none
212.5.7.9   9.180.180.89     17 137> 137 1 x00->x00 PASS none
212.5.7.8   212.1.2.2        1 T:x00 C:x00    1 x00->x00 PASS none
rtr-b DS Console>DSCACHE STATS
Source Address to list []?
Destination Address to list []?
Source      Destination      Pro ProtocolInf Net Tos      RxPkts  RxBytes
212.5.7.9   212.4.4.1        1 T:x08 C:x00    1 x00      4        240
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x30     15       1260
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x38    149     12516
212.4.4.1   212.5.7.9        1 T:x00 C:x00    2 xB8      4        240
212.1.2.2   212.5.7.8        1 T:x08 C:x00    2 x28     15       1260
212.5.7.9   9.180.180.66     17 138> 138 1 x00      2        520
212.5.7.9   9.180.180.89     17 137> 137 1 x00     96       9162
212.5.7.8   212.1.2.2        1 T:x00 C:x00    1 x00    179     15036
rtr-b DS Console>

```

There are several commands to view the DS details in the IP flow cache. **ds actions** shows the source and destination IP addresses, the key fields in the IP header, which interface the packet was received on (Net), what the incoming and outgoing DS-byte (TOS in/out) was and the stream ID. The key fields in the IP header are either (a) TCP/UDP port number, or (b) ICMP type and code field, or (c) fragment number, since fragments don't contain either (a) or (b). This command shows what streams have been created for what traffic, and is very useful to confirm that the DS byte was modified as expected.

The **dscache stats** command shows the statistics for packets for either all addresses or a specific source to a specific destination. The Pro, Protocol Inf and Net information is the same as for the **ds actions** command.



## Monitoring DiffServ - 6



```
rtr-b DS Console>DSCACHE ORDER
Order Source          Destination          Pro ProtocolInf Net Tos
  1 212.5.7.9          9.180.180.89        17 137> 137 1 x00
  2 212.5.7.8          212.1.2.2           1 T:x00 C:x00 1 x00
  3 212.1.2.2          212.5.7.8           1 T:x08 C:x00 2 x38
  4 212.1.2.2          212.5.7.8           1 T:x08 C:x00 2 x30
  5 212.1.2.2          212.5.7.8           1 T:x08 C:x00 2 x28
  6 212.5.7.9          9.180.180.66        17 138> 138 1 x00
  7 212.4.4.1          212.5.7.9           1 T:x00 C:x00 2 xB8
  8 212.5.7.9          212.4.4.1           1 T:x08 C:x00 1 x00
rtr-b DS Console>DSCACHE NEXTHOP
Source Address to list []?
Destination Address to list []?
Source          Destination          Pro ProtocolInf Net Tos NextHop
212.5.7.9       212.4.4.1           1 T:x08 C:x00 1 x00 212.1.2.2 (PPP/2)
212.1.2.2       212.5.7.8           1 T:x08 C:x00 2 x30 212.1.3.2 (PPP/1)
212.1.2.2       212.5.7.8           1 T:x08 C:x00 2 x38 212.1.3.2 (PPP/1)
212.4.4.1       212.5.7.9           1 T:x00 C:x00 2 xB8 212.1.3.2 (PPP/1)
212.1.2.2       212.5.7.8           1 T:x08 C:x00 2 x28 212.1.3.2 (PPP/1)
212.5.7.9       9.180.180.66        17 138> 138 1 x00 212.1.2.2 (PPP/2)
212.5.7.9       9.180.180.89        17 137> 137 1 x00 212.1.2.2 (PPP/2)
212.5.7.8       212.1.2.2           1 T:x00 C:x00 1 x00 212.1.2.2 (PPP/2)
rtr-b DS Console>
```

The **dscache order** command displays the order in which entries were added to the cache. The **dscache nexthop** command is very useful for determining the route of the packet.

► c) view streams

```

rtr-b DS Console>LIST STREAM PACKET-STATS
IN Network number : 2
OUT Network number : 1
At interface 2, 6 in-streams; clock=65435 sec.
Streams from net 2 to net 1:
Id  t I/o q  allo/cur(K)  tot pkt  tot Kby  pkt snt  Kby snt  ovr snt  buf drp pol drp
-----
(20-af)
26  D  in  0.0/ 0.0      191      16      191      16      3      0
    o-q2 0.0/ 0.0      191      16      3      0
(b8)
25  D  in  0.0/ 0.0     4145     2247     4145     2247     7      0
    o-q1 0.0/ 0.0     4145     2247     4145     2247     7      0
(-)
8   B  in  0.0/ 0.0      0      0      0      0      0      0
    o-q2 2.8/ 0.0      0      0      0      0      0      0
(-)
2   C  in  0.0/ 0.0      0      0      0      0      0      0
    o-q2 1.4/ 0.0      2193     297     2193     297     0      0
rtr-b DS Console>

```

If you want to view the status of the streams for a particular ingress-egress pair then use the **list streams** command. This shows (from left to right) the stream ID, the type of stream (D = DiffServ, B = best effort or C = control), which queue the traffic is being handled by (q1=premium), the buffer allocation and current used, the total number and Kbytes received, the number and Kbytes of packets sent, how many packets were oversent (i.e. exceeded the number of buffers defined in the profile), how many packets were dropped and how many packets were dropped by the EF policer.

```

rtr-b DS Console>LIST STREAM METER-MARK
IN Network number : 2
OUT Network number : 1
At interface 2, 6 in-streams; clock=65443 sec.
Streams from net 2 to net 1:
  Id  t I/o q  pkt snt  buf drp  mrk g  mrk y  mrk r  g->y  g->r  y->r
  ---- - - - - -
(20-af)
  26  D  in      191      0
      o-q2    191      0    25    15    151    1    1    1
rtr-b DS Console>LIST STREAM POLICE-PARA
IN Network number : 2
OUT Network number : 1
At interface 2, 6 in-streams; clock=65454 sec.
Streams from net 2 to net 1:
  Id  t I/o q  TR/CIR  TBS/CBS  PIR  EBS/PBS  pol typ
      in bps  in bytes  in bps  in bytes
  ---- - - - - -
(20-af)
  26  D  in
      o-q2          1 4294967006          0 4294967006  SRCA
(b8)
  25  D  in
      o-q1    49229          5225
rtr-b DS Console>

```

Two new commands in V3.4 are **list stream meter-marker** and **list stream police-para**. The meter-marker output shows how many packets have been marked green, yellow and red, and how many have been changed from green to yellow, green to red and yellow to red. Values will only be shown for change if a colour aware marker is used.

The police-para command shows configuration information of the policers.



## Bibliography



- ✓ RFC1812 : Requirements for IPv4 Routers
  - ✓ RFC 2475 : An Architecture for Differentiated Services
  - ✓ RFC 2474 : Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers
  - ✓ RFC2597 : Assured Forwarding PHB Group
  - ✓ RFC2598 : An Expedited Forwarding PHB
  - ✓ draft-heinanen-diffserv-srtcm : A Single Rate Three Color Marker
  - ✓ draft-heinanen-diffserv-trtcm : A Two Rate Three Color Marker
  - ✓ draft-bernet-diffedge : Requirements of Diff-serv Boundary Routers
  - ✓ draft-ietf-diffserv-framework : A Framework for Differentiated Services
  - ✓ draft-guerin-aggreg-rsvp : Aggregating RSVP-based QoS Requests
  - ✓ draft-ietf-diffserv-rsvp : A Framework for Use of RSVP with Diff-Serv Networks
  - ✓ draft-nichols-dsopdef : Differentiated Services Operational Model and Definitions
- 

In the making of this presentation, the above RFCs and drafts were used. IBM routers currently support all the listed RFCs, draft-bernet-diffedge and draft-nichols-dsopdef. They also are well aligned with draft-ietf-diffserv-rsvp for controlled load.