DiffServ Applied to Real-time Transports
Internet-Draft
Intended status: Informational
Expires: March 2, 2015

D. Black, Ed.
EMC
P. Jones
Cisco

August 29, 2014

Differentiated Services (DiffServ) and Real-time Communication
draft-ietf-dart-dscp-rtp-05

Abstract

This memo describes the interaction between Differentiated Services (DiffServ) network quality of service (QoS) functionality and real-time network communication, including communication based on the Real-time Transport Protocol (RTP). DiffServ is based on network nodes applying different forwarding treatments to packets whose IP headers are marked with different DiffServ Code Points (DSCPs). WebRTC applications, as well as some conferencing applications, have begun using the Session Description Protocol (SDP) bundle negotiation mechanism to send multiple traffic streams with different QoS requirements using the same network 5-tuple. Use of different DSCPs to obtain different QoS treatments within a single network 5-tuple, the results (e.g., reordering) may cause transport protocol interactions, particularly with congestion control functionality. In addition, DSCP markings may be changed or removed between the traffic source and destination. This memo covers the implications of these DiffServ aspects for real-time network communication, including WebRTC.

Status of This Memo

This Internet-Draft is submitted in full conformance with the provisions of BCP 78 and BCP 79.

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF). Note that other groups may also distribute working documents as Internet-Drafts. The list of current Internet-Drafts is at http://datatracker.ietf.org/drafts/current/.

Internet-Drafts are draft documents valid for a maximum of six months and may be updated, replaced, or obsoleted by other documents at any time. It is inappropriate to use Internet-Drafts as reference material or to cite them other than as "work in progress."

This Internet-Draft will expire on March 2, 2015.

Table of Contents

1. Introduction

This memo describes the interactions between Differentiated Services (DiffServ) network quality of service (QoS) functionality [RFC2475] and real-time network communication, including communication based on the Real-time Transport Protocol (RTP) [RFC3550]. DiffServ is based on network nodes applying different forwarding treatments to packets whose IP headers are marked with different DiffServ Code Points (DSCPs)[RFC2474]. In the past, distinct RTP streams have been sent

over different transport level flows, sometimes multiplexed with RTCP. WebRTC applications, as well as some conferencing applications, are now using the Session Description Protocol (SDP) [RFC4566] bundle negotiation mechanism [I-D.ietf-mmusic-sdp-bundle-negotiation] to send multiple traffic streams with different QoS requirments using the same network 5-tuple. Use of different DSCPs to obtain different QoS treatments within a single network 5-tuple, the results (e.g., reordering) may cause transport protocol interactions, particularly with congestion control functionality. In addition, DSCP markings may be changed or removed between the traffic source and destination. This memo covers the implications of these DiffServ aspects for real-time network communication, including WebRTC traffic [I-D.ietf-rtcweb-overview].

The memo is organized as follows. Background is provided in Section 2 on real time communications and Section 3 on Differentiated Services. Section 4 describes some examples of DiffServ usage with real time communications. Section 5 explains how use of DiffServ features interacts with both transport and real time communications protocols and Section 6 provides guidance on DiffServ feature usage to control undesired interactions. Security considerations are discussed in Section 8 (Section 7 is an empty IANA Considerations section).

2. Real Time Communications

Real-time communications enables communication in real time over an IP network using voice, video, text, content sharing, etc. It is possible to use one or more of these modalities in parallel to provide a rich communication experience.

A simple example of real-time communications is a voice call placed over the Internet where an audio stream is transmitted in each direction between two users. A more complex example is an immersive videoconferencing system that has multiple video screens, multiple cameras, multiple microphones, and some means of sharing content. For such complex systems, there may be multiple media and non-media streams transmitted via a single IP address and port or via multiple IP addresses and ports.

2.1. RTP Background

The most common protocol used for real time media is the Real-Time Transport Protocol (RTP) [RFC3550]. RTP defines a common encapsulation format and handling rules for real-time data transmitted over the Internet. Unfortunately, RTP terminology usage has been inconsistent. For example, the document on RTP grouping terminology [I-D.ietf-avtext-rtp-grouping-taxonomy] observes that:

RFC 3550 [RFC3550] uses the terms media stream, audio stream, video stream and streams of (RTP) packets interchangeably.

Terminology in this memo is based on that RTP grouping terminology document with the following terms being of particular importance (see that terminology document for full definitions):

Source Stream: A reference clock synchronized, time progressing, digital media stream.

RTP Stream: A stream of RTP packets containing media data, which may be source data or redundant data. The RTP Stream is identified by an RTP synchronization source (SSRC) belonging to a particular RTP session.

In addition, this memo follows [RFC3550] in using the term "SSRC" to designate both the identifier of an RTP stream and the entity that sends that RTP stream.

Media encoding and packetization of a source stream results in a source RTP stream plus zero or more redundancy RTP streams that provide resilience against loss of packets from the source RTP stream [I-D.ietf-avtext-rtp-grouping-taxonomy]. Redundancy information may also be carried in the same RTP stream as the encoded source stream, e.g., see Section 7.2 of [RFC5109]. With most applications, a single media type (e.g., audio) is transmitted within a single RTP session. However, it is possible to transmit multiple, distinct source streams over the same RTP session as one or more individual RTP streams. This is referred to as RTP multiplexing. In addition, an RTP stream may contain multiple source streams, e.g., components or programs in an MPEG Transport Stream [H.222.0].

The number of source streams and RTP streams in an overall real-time interaction can be surprisingly large. In addition to a voice source stream and a video source stream, there could be separate source streams for each of the cameras or microphones on a videoconferencing system. As noted above, there might also be separate redundancy RTP streams that provide protection to a source RTP stream, using techniques such as Forward Error Correction. Another example is simulcast transmission, where a video source stream can be transmitted as high resolution and low resolution RTP streams at the same time. In this case, a media processing function might choose to send one or both RTP streams onward to a receiver based on bandwidth availability or who the active speaker is in a multipoint conference. Lastly, a transmitter might send the same media content concurrently as two RTP streams using different encodings (e.g., video encoded as VP8 in parallel with H.264) to allow a media processing function to

select a media encoding that best matches the capabilities of the receiver.

For the WebRTC protocol suite [I-D.ietf-rtcweb-transports], an individual source stream is a MediaStreamTrack, and a MediaStream contains one or more MediaStreamTracks [W3C.WD-mediacapture-streams-20130903]. A MediaStreamTrack is transmitted as a source RTP stream plus zero or more redundant RTP streams, so a MediaStream that consists of one MediaStreamTrack is transmitted as a single source RTP stream plus zero or more redundant RTP streams. For more information on use of RTP in WebRTC, see [I-D.ietf-rtcweb-rtp-usage].

RTP is usually carried over a datagram protocol, such as UDP[RFC0768], UDP-Lite [RFC3828] or DCCP [RFC4340]; UDP is most commonly used, but a non-datagram protocol (e.g., TCP) may also be used. Transport protocols other than UDP or UDP-Lite may also be used to transmit real-time data or near-real-time data. For example, SCTP [RFC4960] can be utilized to carry application sharing or whiteboarding information as part of an overall interaction that includes real-time media. These additional transport protocols can be multiplexed with an RTP session via UDP encapsulation, thereby using a single pair of UDP ports.

The WebRTC protocol suite encompasses a number of forms of multiplexing:

1. Individual source streams are carried in one or more individual RTP streams. These RTP streams can be multiplexed onto a single transport-layer flow or sent as separate transport-layer flows. This memo only considers the case where the RTP streams are to be multiplexed onto a single transport-layer flow, forming a single RTP session as described in [RFC3550];

2. The RTP Control Protocol (RTCP) (see [RFC3550]) may be multiplexed onto the same transport-layer flow as the RTP streams with which it is associated, as described in [RFC5761] or it may be sent on a separate transport-layer flow;

3. An RTP session could be multiplexed with a single SCTP association over DTLS and with both STUN [RFC5389] and TURN [RFC5766] traffic into a single transport-layer flow as described in [RFC5764] with the updates in [I-D.petithuguenin-avtcore-rfc5764-mux-fixes]. The STUN [RFC5389] and TURN [RFC5766] protocols provide NAT/FW (Network Address Translator / Firewall) traversal and port mapping.

The resulting transport layer flow is identified by a network 5-tuple, i.e., a combination of two IP addresses (source and destination), two ports (source and destination), and the transport protocol used (e.g., UDP). SDP bundle negotiation restrictions [I-D.ietf-mmusic-sdp-bundle-negotiation] limit WebRTC to using at most a single DTLS session per network 5-tuple. In contrast to WebRTC use of a single SCTP association with DTLS, multiple SCTP associations can be directly multiplexed over a single UDP 5-tuple as specified in [RFC6951].

The STUN and TURN protocols were originally designed for use of UDP, however, TURN has been extended to use TCP as a transport for situations in which UDP does not work [RFC6062]. When TURN selects use of TCP, the entire real-time communications session is carried over a single TCP 5-tuple.

For IPv6, addition of the flow label [RFC6437] to network 5-tuples results in network 6-tuples, but in practice, use of a flow label is unlikely to result in a finer-grain traffic subset than the corresponding network 5-tuple (e.g., the flow label is likely to represent the combination of two ports with use of the UDP protocol). For that reason, discussion in this draft focuses on UDP 5-tuples.

2.2. RTP Multiplexing

Section 2.1 explains how source streams can be multiplexed in a single RTP session, which can in turn be multiplexed over UDP with packets generated by other transport protocols. This section provides background on why this level of multiplexing is desirable. The rationale in this section applies both to multiplexing of source streams in a single RTP session and multiplexing of an RTP session with traffic from other transport protocols via UDP encapsulation.

Multiplexing reduces the number of ports utilized for real-time and related communication in an overall interaction. While a single endpoint might have plenty of ports available for communication, this traffic often traverses points in the network that are constrained on the number of available ports or whose performance degrades as the number of ports in use increases. A good example is a Network Address Translator and Firewall (NAT/FW) device sitting at the network edge. As the number of simultaneous protocol sessions increases, so does the burden placed on these devices to provide port mapping.

Another reason for multiplexing is to help reduce the time required to establish bi-directional communication. Since any two communicating users might be situated behind different NAT/FW devices, it is necessary to employ techniques like STUN and TURN

along with ICE [RFC5245] to get traffic to flow between the two devices [I-D.ietf-rtcweb-transports]. Performing the tasks required by these protocols takes time, especially when multiple protocol sessions are involved. While tasks for different sessions can be performed in parallel, it is nonetheless necessary for applications to wait for all sessions to be opened before communication between two users can begin. Reducing the number of STUN/ICE/TURN steps reduces the likelihood of loss of a packet for one of these protocols; any such loss adds delay to setting up a communication session. Further, reducing the number of STUN/ICE/TURN tasks places a lower burden on the STUN and TURN servers.

Multiplexing may reduce the complexity and resulting load on an endpoint. A single instance of STUN/ICE/TURN is simpler to execute and manage than multiple instances STUN/ICE/TURN operations happening in parallel, as the latter require synchronization and create more complex failure situations that have to be cleaned up by additional code.

3. Differentiated Services (DiffServ)

The DiffServ architecture [RFC2475][RFC4594] is intended to enable scalable service discrimination in the Internet without requiring each node in the network to store per-flow state and participate in per-flow signaling. The services may be end-to-end or within a network; they include both those that can satisfy quantitative performance requirements (e.g., peak bandwidth) and those based on relative performance (e.g., "class" differentiation). Services can be constructed by a combination of well-defined building blocks deployed in network nodes that:

- classify traffic and set bits in an IP header field at network boundaries or hosts,
- use those bits to determine how packets are forwarded by the nodes inside the network, and
- condition the marked packets at network boundaries in accordance with the requirements or rules of each service.

Traffic conditioning may include changing the DSCP in a packet (remarking it), delaying the packet (as a consequence of traffic shaping) or dropping the packet (as a consequence of traffic policing).

A network node that supports DiffServ includes a classifier that selects packets based on the value of the DS field in IP headers (the DiffServ codepoint or DSCP), along with buffer management and packet

scheduling mechanisms capable of delivering the specific packet forwarding treatment indicated by the DS field value. Setting of the DS field and fine-grain conditioning of marked packets need only be performed at network boundaries; internal network nodes operate on traffic aggregates that share a DS field value, or in some cases, a small set of related values.

The DiffServ architecture[RFC2475] maintains distinctions among:

- the QoS service provided to a traffic aggregate,
- the conditioning functions and per-hop behaviors (PHBs) used to realize services,
- the DSCP in the IP header used to mark packets to select a per-hop behavior, and
- the particular implementation mechanisms that realize a per-hop behavior.

This memo focuses on PHBs and the usage of DSCPs to obtain those behaviors. In a network node's forwarding path, the DSCP is used to map a packet to a particular forwarding treatment, or per-hop behavior (PHB) that specifies the forwarding treatment.

The specification of a PHB describes the externally observable forwarding behavior of a network node for network traffic marked with a DSCP that selects that PHB. In this context, "forwarding behavior" is a general concept - for example, if only one DSCP is used for all traffic on a link, the observable forwarding behavior (e.g., loss, delay, jitter) will often depend only on the loading of the link. To obtain useful behavioral differentiation, multiple traffic subsets are marked with different DSCPs for different PHBs for which node resources such as buffer space and bandwidth are allocated. PHBs provide the framework for a DiffServ network node to allocate resources to traffic subsets, with network-scope differentiated services constructed on top of this basic hop-by-hop resource allocation mechanism.

The codepoints (DSCPs) may be chosen from a small set of fixed values (the class selector codepoints), or from a set of recommended values defined in PHB specifications, or from values that have purely local meanings to a specific network that supports DiffServ; in general, packets may be forwarded across multiple such networks between source and destination.

The mandatory DSCPs are the class selector code points as specified in [RFC2474]. The class selector codepoints (CS0-CS7) extend the

deprecated concept of IP Precedence in the IPv4 header; three bits are added, so that the class selector DSCPs are of the form 'xxx000'. The all-zero DSCP ('000000' or CS0) is always assigned to a Default PHB that provides best-effort forwarding behavior and the remaining class selector code points are intended to provide relatively better per-hop-forwarding behavior in increasing numerical order, but:

- There is no requirement that any two adjacent class selector codepoints be assigned to different PHBs; adjacent class selector codepoints may use the same pool of resources on each network node in some networks. This generalizes to ranges of class selector codepoints, but with limits - for example CS6 and CS7 are often used for network control (e.g., routing) traffic [RFC4594] and hence are likely to provide better forwarding behavior under network load to prioritize network recovery from disruptions.

- CS1 ('001000') was subsequently designated as the recommended codepoint for the Lower Effort (LE) PHB [RFC3662]. An LE service forwards traffic with "lower" priority than best effort and can be "starved" by best effort and other "higher" priority traffic. Not all networks offer an LE service, hence traffic marked with the CS1 DSCP may not receive lower effort forwarding; such traffic may be forwarded with a different PHB (e.g., the Default PHB), remarked to another DSCP (e.g., CS0) and forwarded accordingly, or dropped. See [RFC3662] for further discussion of the LE PHB and service.

One cannot rely upon different class selector codepoints providing differentiated services or upon the presence of an LE service that is selected by the CS1 DSCP. There is no effective way for a network endpoint to determine which PHBs are selected by the class selector codepoints or whether the CS1 DSCP selects an LE service on a specific network, let alone end-to-end. Packets marked with the CS1 DSCP may be forwarded with best effort service or another "higher" priority service, see [RFC2474].

3.1. Diffserv PHBs (Per-Hop Behaviors)

Although Differentiated Services is a general architecture that may be used to implement a variety of services, three fundamental forwarding behaviors (PHBs) have been defined and characterized for general use. These are:

1. Default Forwarding (DF) for elastic traffic [RFC2474]. The Default PHB is always selected by the all-zero DSCP and provides best-effort forwarding.

2. Assured Forwarding (AF) [RFC2597] to provide differentiated service to elastic traffic. Each instance of the AF behavior consists of three PHBs that differ only in drop precedence, e.g., AF11, AF12 and AF13; such a set of three AF PHBs is referred to as an AF class, e.g., AF1x. There are four defined AF classes, AF1x through AF4x, with higher numbered classes intended to receive better forwarding treatment than lower numbered classes.

3. Expedited Forwarding (EF) [RFC3246] intended for inelastic traffic. Beyond the basic EF PHB, the VOICE-ADMIT PHB [RFC5865] is an admission controlled variant of the EF PHB. Both of these PHBs are based on pre-configured limited forwarding capacity; traffic in excess of that capacity is expected to be dropped.

3.2. Traffic Classifiers and DSCP Remarking

DSCP markings are not end-to-end in general. Each network can make its own decisions about what PHBs to use and which DSCP maps to each PHB. While every PHB specification includes a recommended DSCP, and RFC 4594 [RFC4594] recommends their end-to-end usage, there is no requirement that every network support any PHBs or use any specific DSCPs, with the exception of the support requirements for the class selector codepoints (see RFC 2474 [RFC2474]). When DiffServ is used, the edge or boundary nodes of a network are responsible for ensuring that all traffic entering that network conforms to that network's policies for DSCP and PHB usage, and such nodes may change DSCP markings on traffic to achieve that result. As a result, DSCP remarking is possible at any network boundary, including the first network node that traffic sent by a host encounters. Remarking is also possible within a network, e.g., for traffic shaping.

DSCP remarking is part of traffic conditioning; the traffic conditioning functionality applied to packets at a network node is determined by a traffic classifier [RFC2475]. Edge nodes of a DiffServ network classify traffic based on selected packet header fields; typical implementations do not look beyond the traffic's network 5-tuple in the IP and transport protocol headers (e.g., for SCTP or RTP enapsulated in UDP, header-based classification is unlikely to look beyond the outer UDP header). As a result, when multiple DSCPs are used for traffic that shares a network 5-tuple, remarking at a network boundary may result in all of the traffic being forwarded with a single DSCP, thereby removing any differentiation within the network 5-tuple downstream of the remarking location. Network nodes within a DiffServ network generally classify traffic based solely on DSCPs, but may perform finer grain traffic conditioning similar to that performed by edge nodes.

So, for two arbitrary network endpoints, there can be no assurance that the DSCP set at the source endpoint will be preserved and presented at the destination endpoint. Rather, it is quite likely that the DSCP will be set to zero (e.g., at the boundary of a network operator that distrusts or does not use the DSCP field) or to a value deemed suitable by an ingress classifier for whatever network 5-tuple it carries.

In addition, remarking may remove application-level distinctions in forwarding behavior - e.g., if multiple PHBs within an AF class are used to distinguish different types of frames within a video RTP stream, token-bucket-based remarkers operating in Color-Blind mode (see [RFC2697] and [RFC2698] for examples) may remark solely based on flow rate and burst behavior, removing the drop precedence distinctions specified by the source.

Backbone and other carrier networks may employ a small number of DSCPs (e.g., less than half a dozen) to manage a small number of traffic aggregates; hosts that use a larger number of DSCPs can expect to find that much of their intended differentiation is removed by such networks. Better results may be achieved when DSCPs are used to spread traffic among a smaller number of DiffServ-based traffic subsets or aggregates, see [I-D.geib-tsvwg-diffserv-intercon] for one proposal. This is of particular importance for MPLS-based networks due to the limited size of the Traffic Class (TC) field in an MPLS label [RFC5462] that is used to carry DiffServ information and the use of that TC field for other purposes, e.g., ECN [RFC5129]. For further discussion on use of DiffServ with MPLS, see [RFC3270] and [RFC5127].

4. Examples

For real-time communications, one might want to mark the audio packets using EF and the video packets as AF41. However, in a video conference receiving the audio packets significantly ahead of the video is not useful because lip sync is necessary between audio and video. It may still be desirable to send audio with a PHB that provides better service, because more reliable arrival of audio helps assure smooth audio rendering, which is often more important than fully faithful video rendering. There are also limits, as some devices have difficulties in synchronizing voice and video when packets that need to be rendered together arrive at significantly different times. It makes more sense to use different PHBs when the audio and video source streams do not share a strict timing relationship. For example, video content may be shared within a video conference via playback, perhaps of an unedited video clip that is intended to become part of a television advertisement. Such content sharing video does not need precise synchronization with

video conference audio, and could use a different PHB, as content sharing video is more tolerant to jitter, loss, and delay.

Within a layered video RTP stream, ordering of frame communication is preferred, but importance of frame types varies, making use of PHBs with different drop precedences appropriate. For example, I-frames that contain an entire image are usually more important than P-frames that contain only changes from the previous image because loss of a P-frame (or part thereof) can be recovered (at the latest) via the next I-frame, whereas loss of an I-frame (or part thereof) may cause rendering problems for all of the P-frames that depend on the missing I-frame. For this reason, it is appropriate to mark I-frame packets with a PHB that has lower drop precedence than the PHB used for P-frames, as long as the PHBs preserve ordering among frames (e.g., are in a single AF class) - AF41 for I-frames and AF43 for P-frames is one possibility. Additional spatial and temporal layers beyond the base video layer could also be marked with higher drop precedence than the base video layer, as their loss reduces video quality, but does not disrupt video rendering.

Additional RTP streams in a real-time communication interaction could be marked with CS0 and carried as best effort traffic. One example is real-time text transmitted as specified in RFC 4103 [RFC4103]. Best effort forwarding suffices because such real-time text has loose timing requirements; RFC 4103 recommends sending text in chunks every 300ms. Such text is technically real-time, but does not need a PHB promising better service than best effort, in contrast to audio or video.

A WebRTC application may use one or more RTP streams, as discussed above. In addition, it may use an SCTP-based data channel [I-D.ietf-rtcweb-data-channel] whose QoS treatment depends on the nature of the application. For example, best effort treatment of data channels is likely to suffice for messaging, shared white board, and guided browsing applications, whereas latency-sensitive games might desire better QoS for their data channels.

5. DiffServ Interactions

5.1. DiffServ, Reordering and Transport Protocols

Transport protocols provide data communication behaviors beyond those possible at the IP layer. An important example is that TCP [RFC0793] provides reliable in-order delivery of data with congestion control. SCTP [RFC4960] provides additional properties such as preservation of message boundaries, and the ability to avoid head-of-line blocking that may occur with TCP.

In contrast, UDP [RFC0768] is a basic unreliable datagram protocol that provides port-based multiplexing and demultiplexing on top of IP. Two other unreliable datagram protocols are UDP-Lite [RFC3828], a variant of UDP that may deliver partially corrupt payloads when errors occur, and DCCP [RFC4340], which provides a range of congestion control modes for its unreliable datagram service.

Transport protocols that provide reliable delivery (e.g., TCP, SCTP) are sensitive to network reordering of traffic. When a protocol that provides reliable delivery receives a packet other than the next expected packet, the protocol usually assumes that the expected packet has been lost and respond with a retransmission request for that packet. In addition, congestion control functionality in transport protocols (including DCCP) usually infers congestion when packets are lost. This creates additional sensitivity to significant network packet reordering, as such reordering may be (mis-)interpreted as loss of the out-of-order packets, causing a congestion control response.

This sensitivity to reordering remains even when ECN [RFC3168] is in use, as ECN receivers are required to treat missing packets as potential indications of congestion, because:

- Severe congestion may cause ECN-capable network nodes to drop packets, and

- ECN traffic may be forwarded by network nodes that do not support ECN and hence drop packets to indicate congestion.

Congestion control is an important aspect of the Internet architecture, see [RFC2914] for further discussion.

In general, marking packets with different DSCPs results in different PHBs being applied at nodes in the network, making reordering possible due to use of different pools of forwarding resources for each PHB. This should not be done for current transport protocols within a single network 5-tuple, with the exception of UDP and UDP-Lite.

When PHBs that allow reordering are mixed within a single network 5-tuple, the effect is to mix QoS-based traffic classes within the scope of a single transport connection or association. Such QoS-based traffic classes receive different network QoS treatments and hence use different pools of network resources. Transport protocol support for multiple QoS-based traffic classes within a single network 5-tuple adds complexity to congestion-controlled transport protocols by comparison to current designs because it requires that network congestion information for each QoS-based traffic class be

disambiguated so that congestion control is managed separately for each such traffic class. Traffic in different QoS-based traffic classes may use different paths through the network; this complicates path integrity checking in connection- or association-based protocols, as those paths may fail independently.

The primary example where usage of multiple PHBs does not allow reordering within a single network 5-tuple is use of PHBs from a single AF class (e.g., AF1x). Traffic reordering within the scope of a network 5-tuple that uses a single PHB or AF class may occur for other transient reasons (e.g., routing changes or ECMP rebalancing).

Reordering also affects other forms of congestion control, such as techniques for RTP congestion control that were under development when this memo was published, see [I-D.ietf-rmcat-cc-requirements] for requirements. These techniques prefer use of a common (coupled) congestion controller for RTP streams between the same endpoints to reduce packet loss and delay by reducing competition for resources at any shared bottleneck.

Shared bottlenecks can be detected via techniques such as correlation of one-way delay measurements across RTP streams. An alternate approach is to assume that the set of packets on a single network 5-tuple marked with DSCPs that do not allow reordering will utilize a common network path and common forwarding resources at each network node. Under that assumption, any bottleneck encountered by such packets is shared among all of them, making it safe to use a common (coupled) congestion controller, see [I-D.welzl-rmcat-coupled-cc]. This is not a safe assumption when the packets involved are marked with DSCP values that allow reordering because a bottleneck may not be shared among all such packets (e.g., if the DSCPs result in use of different queues at a network node, only one of which is a bottleneck).

UDP and UDP-Lite are not sensitive to reordering in the network, because they do not provide reliable delivery or congestion control. On the other hand, when used to encapsulate other protocols (e.g., as UDP is used by WebRTC, see Section 2.1), the reordering considerations for the encapsulated protocols apply. For the specific usage of UDP by WebRTC, every encapsulated protocol (i.e., RTP, SCTP and TCP) is sensitive to reordering as further discussed in this memo. In addition, [RFC5405] provides general guidelines for use of UDP (and UDP-Lite); the congestion control guidelines in that document apply to protocols encapsulated in UDP (or UDP-Lite).

5.2. DiffServ, Reordering and Real-Time Communication

Real-time communications are also sensitive to network reordering of packets. Such reordering may lead to spurious NACK generation and unneeded retransmission, as is the case for reliable delivery protocols (see Section 5.1). The degree of sensitivity depends on protocol or stream timers, in contrast to reliable delivery protocols that usually react to all reordering.

Receiver jitter buffers have important roles in the effect of reordering on real time communications:

- Minor packet reordering that is contained within a jitter buffer usually has no effect on rendering of the received RTP stream because packets that arrive out of order are retrieved in order from the jitter buffer for rendering.

- Packet reordering that exceeds the capacity of a jitter buffer can cause user-perceptible quality problems (e.g., glitches, noise) for delay sensitive communication, such as interactive conversations for which small jitter buffers are necessary to preserve human perceptions of real-time interaction. Interactive real-time communication implementations often discard data that is sufficiently late that it cannot be rendered in source stream order, making retransmission counterproductive. For this reason, implementations of interactive real-time communication often do not use retransmission.

- In contrast, replay of recorded media can tolerate significantly longer delays than interactive conversations, so replay is likely to use larger jitter buffers than interactive conversations. These larger jitter buffers increase the tolerance of replay to reordering by comparison to interactive conversations. The size of the jitter buffer imposes an upper bound on replay tolerance to reordering, but does enable retransmission to be used when the jitter buffer is significantly larger than the amount of data that can be expected to arrive during the round-trip latency for retransmission.

Network packet reordering has no effective upper bound, and can exceed the size of any reasonable jitter buffer. In practice, the size of jitter buffers for replay is limited by external factors such as the amount of time that a human is willing to wait for replay to start.

5.3. Drop Precedence and Transport Protocols

Packets within the same network 5-tuple that use PHBs within a single AF class can be expected to draw upon the same forwarding resources on network nodes (e.g., use the same router queue) and hence use of multiple drop precedences within an AF class is not expected to cause latency variation. When PHBs within a single AF class are mixed within a flow, the resulting overall likelihood that packets will be dropped from that flow is a mix of the drop likelihoods of the PHBs involved.

There are situations in which drop precedences should not be mixed. A simple example is that there is little value in mixing drop precedences within a TCP connection, because TCP's ordered delivery behavior results in any drop requiring the receiver to wait for the dropped packet to be retransmitted. Any resulting delay depends on the RTT and not the packet that was dropped. Hence a single DSCP should be used for all packets in a TCP connection.

As a consequence, when TCP is selected for NAT/FW traversal (e.g., by TURN), a single DSCP should be used for all traffic on that TCP connection. An additional reason for this recommendation is that packetization for STUN/ICE/TURN occurs before passing the resulting packets to TCP; TCP resegmentation may result in a different packetization on the wire, breaking any association between DSCPs and specific data to which they are intended to apply.

SCTP [RFC4960] differs from TCP in a number of ways, including the ability to deliver messages in an order that differs from the order in which they were sent and support for unreliable streams. However, SCTP performs congestion control and retransmission across the entire association, and not on a per-stream basis. Although there may be advantages to using multiple drop precedence across SCTP streams or within an SCTP stream that does not use reliable ordered delivery, there is no practical operational experience in doing so (e.g., the SCTP sockets API [RFC6458] does not support use of more than one DSCP for an SCTP association). As a consequence, the impacts on SCTP protocol and implementation behavior are unknown and difficult to predict. Hence a single DSCP should be used for all packets in an SCTP association, independent of the number or nature of streams in that association. Similar reasoning applies to a DCCP connection; a single DSCP should be used because the scope of congestion control is the connection and there is no operational experience with using more than one DSCP. This recommendation may be revised in the future if experiments, analysis and operational experience provide compelling reasons to change it.

Guidance on transport protocol design and implementation to provide support for use of multiple PHBs and DSCPs in a transport protocol connection (e.g., DCCP) or transport protocol association (e.g., SCTP) is out of scope for this memo.

5.4. DiffServ and RTCP

The RTP Control Protocol (RTCP) [RFC3550] is used with RTP to monitor quality of service and convey information about RTP session participants. A sender of RTCP packets that also sends RTP packets (i.e., originates an RTP stream) should use the same DSCP marking for both types of packets. If an RTCP sender doesn't send any RTP packets, it should mark its RTCP packets with the DSCP that it would use if it did send RTP packets with media similar to the RTP traffic that it receives. If the RTCP sender uses or would use multiple DSCPs that differ only in drop precedence for RTP, then it should use the DSCP with the least likelihood of drop for RTCP to increase the likelihood of RTCP packet delivery.

If the SDP bundle extension [I-D.ietf-mmusic-sdp-bundle-negotiation] is used to negotiate sending multiple types of media in a single RTP session, then receivers will send separate RTCP reports for each type of media, using a separate SSRC for each media type; each RTCP report should be marked with the DSCP corresponding to the type of media handled by the reporting SSRC.

This guidance may result in different DSCP markings for RTP streams and RTCP receiver reports about those RTP streams. The resulting variation in network QoS treatment by traffic direction could result in unrepresentative round trip time (RTT) estimates that don't correspond to consistent network QoS treatment in both directions. RTCP receiver reports may be relatively infrequent (e.g., may be sent only once per video frame rendered) and hence the resulting RTT estimates are of limited utility for congestion control (although they have other important uses, see [RFC3550]). For this reason, it is not important that RTCP receiver reports receive the same network QoS treatment as the RTP stream or streams being reported on.

RTCP multi-stream reporting optimizations for an RTP session [I-D.ietf-avtcore-rtp-multi-stream-optimisation] assume that the RTP streams involved experience the same network behavior, including packet loss. This mechanism is highly inappropriate when the received RTP streams involved use different DSCPs, even if the corresponding PHBs differ solely in drop precedence.

6. Guidelines

The only use of multiple standardized PHBs and DSCPs that prevents network reordering among packets marked with different DSCPs is use of PHBs within a single AF class. All other uses of multiple PHBs and/or the class selector DSCPs allow network reordering of packets that are marked with different DSCPs. Based on this and the foregoing discussion, the guidelines in this section apply to use of DiffServ with real-time communications.

Applications and other traffic sources:

- Should limit use of DSCPs within a single RTP stream to those whose corresponding PHBs do not allow packet reordering. If this is not done, significant network reordering may overwhelm implementation assumptions about reordering limits, e.g., jitter buffer size, causing poor user experiences, see Section 5.2 above. This guideline applies to all of the RTP streams that are within the scope of a common (coupled) congestion controller when that controller does not use per-RTP-stream measurements for bottleneck detection.

- Should use a single DSCP for RTCP packets, which should be a DSCP used for RTP packets that are or would be sent by that SSRC, see Section 5.4.

- Should use a single DSCP for all packets within a reliable transport protocol session (e.g., TCP connection, SCTP association) or DCCP connection, see Section 5.1 and Section 5.3. For SCTP, this requirement applies across the entire SCTP association, and not just to individual streams within an association. When TURN selects TCP for NAT/FW traversal, this guideline applies to all traffic multiplexed onto that TCP connection, in contrast to use of UDP for NAT/FW traversal.

- May use different DSCPs whose corresponding PHBs allow reordering within a single UDP or UDP-Lite 5-tuple, subject to the above constraints. The service differentiation provided by such usage is unreliable, as it may be removed or changed by DSCP remarking at network boundaries as described in Section 3.2 above.

- Cannot rely on end-to-end preservation of DSCPs as network node remarking can change DSCPs and remove drop precedence distinctions (see Section 3.2 above). For example, if a source uses drop precedence distinctions within an AF class to identify different types of video frames, using those DSCP values at the receiver to identify frame type is inherently unreliable.

- Should limit use of the CS1 codepoint to traffic for which best effort forwarding is acceptable, as network support for use of CS1 to select a "less than best effort" PHB is inconsistent. Further, some networks may treat CS1 as providing "better than best effort" forwarding behavior.

The above guidelines apply independently to each SSRC that sends RTP traffic.

There is no guidance in this memo on how network operators should differentiate traffic. Networks may support all of the PHBs discussed herein, classify EF and AFxx traffic identically, or even remark all traffic to best effort at some ingress points. Nonetheless, it is useful for applications and other traffic sources to provide finer granularity DSCP marking on packets for the benefit of networks that offer QoS service differentiation. A specific example is that traffic originating from a browser may benefit from QoS service differentiation in within-building and residential access networks, even if the DSCP marking is subsequently removed or simplified. This is because such networks and the boundaries between them are likely traffic bottleneck locations (e.g., due to customer aggregation onto common links and/or speed differences among links used by the same traffic).

7. IANA Considerations

This memo includes no request to IANA.

8. Security Considerations

The security considerations for all of the technologies discussed in this memo apply; in particular see the security considerations for RTP in [RFC3550] and DiffServ in [RFC2474] and[RFC2475].

Multiplexing of multiple protocols onto a single UDP 5-tuple via encapsulation has implications for network functionality that monitors or inspects individual protocol flows, e.g., firewalls and traffic monitoring systems. When implementations of such functionality lack visibility into encapsulated traffic (likely for many current implementations), it may be difficult or impossible to apply network security policy and associated controls at a finer granularity than the overall UDP 5-tuple.

Use of multiple DSCPs that allow reordering within an overall real-time communication interaction enlarges the set of network forwarding resources used by that interaction, thereby increasing exposure to resource depletion or failure, independent of whether the underlying cause is benign or malicious. This represents an increase in the

effective attack surface of the interaction, and is a consideration in selecting an appropriate degree of QoS differentiation among the components of the real-time communication interaction.

Use of multiple DSCPs to provide differentiated QoS service may reveal information about the encrypted traffic to which different service levels are provided. For example, DSCP-based identification of RTP streams combined with packet frequency and packet size could reveal the type or nature of the encrypted source streams. The IP header used for forwarding has to be unencrypted for obvious reasons, and the DSCP likewise has to be unencrypted to enable different IP forwarding behaviors to be applied to different packets. The nature of encrypted traffic components can be disguised via encrypted dummy data padding and encrypted dummy packets, e.g., see the discussion of traffic flow confidentiality in [RFC4303]. Encrypted dummy packets could even be added in a fashion that an observer of the overall encrypted traffic might mistake for another encrypted RTP stream.

9. Acknowledgements

This memo is the result of many conversations that have occurred within the dart working group and multiple other working groups in the RAI and Transport areas. Many thanks to Aamer Akhter, Harald Alvestrand, Erin Bournival, Ben Campbell, Brian Carpenter, Keith Drage, Gorry Fairhurst, Ruediger Geib, Cullen Jennings, Jonathan Lennox, Karen Nielsen, Colin Perkins, James Polk, Michael Welzl, Dan York and the dart WG participants for their reviews and comments.

10. References

10.1. Normative References

[I-D.ietf-avtext-rtp-grouping-taxonomy]
Lennox, J., Gross, K., Nandakumar, S., and G. Salgueiro, "A Taxonomy of Grouping Semantics and Mechanisms for Real-Time Transport Protocol (RTP) Sources", draft-ietf-avtext-rtp-grouping-taxonomy-02 (work in progress), June 2014.

[RFC0768] Postel, J., "User Datagram Protocol", STD 6, RFC 768, August 1980.

[RFC0793] Postel, J., "Transmission Control Protocol", STD 7, RFC 793, September 1981.

[RFC2474] Nichols, K., Blake, S., Baker, F., and D. Black, "Definition of the Differentiated Services Field (DS Field) in the IPv4 and IPv6 Headers", RFC 2474, December 1998.

[RFC2475] Blake, S., Black, D., Carlson, M., Davies, E., Wang, Z., and W. Weiss, "An Architecture for Differentiated Services", RFC 2475, December 1998.

[RFC2597] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", RFC 2597, June 1999.

[RFC3246] Davie, B., Charny, A., Bennet, J., Benson, K., Le Boudec, J., Courtney, W., Davari, S., Firoiu, V., and D. Stiliadis, "An Expedited Forwarding PHB (Per-Hop Behavior)", RFC 3246, March 2002.

[RFC3550] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003.

[RFC3662] Bless, R., Nichols, K., and K. Wehrle, "A Lower Effort Per-Domain Behavior (PDB) for Differentiated Services", RFC 3662, December 2003.

[RFC3828] Larzon, L-A., Degermark, M., Pink, S., Jonsson, L-E., and G. Fairhurst, "The Lightweight User Datagram Protocol (UDP-Lite)", RFC 3828, July 2004.

[RFC4340] Kohler, E., Handley, M., and S. Floyd, "Datagram Congestion Control Protocol (DCCP)", RFC 4340, March 2006.

[RFC4960] Stewart, R., "Stream Control Transmission Protocol", RFC 4960, September 2007.

[RFC5405] Eggert, L. and G. Fairhurst, "Unicast UDP Usage Guidelines for Application Designers", BCP 145, RFC 5405, November 2008.

[RFC5865] Baker, F., Polk, J., and M. Dolly, "A Differentiated Services Code Point (DSCP) for Capacity-Admitted Traffic", RFC 5865, May 2010.

[RFC6951] Tuexen, M. and R. Stewart, "UDP Encapsulation of Stream Control Transmission Protocol (SCTP) Packets for End-Host to End-Host Communication", RFC 6951, May 2013.

10.2. Informative References

[H.222.0] ITU-T, "H.222.0 : Information technology - Generic coding of moving pictures and associated audio information", June 2012.

[I-D.geib-tsvwg-diffserv-intercon]
Geib, R., "DiffServ interconnection classes and practice", draft-geib-tsvwg-diffserv-intercon-06 (work in progress), July 2014.

[I-D.ietf-avtcore-rtp-multi-stream-optimisation]
Lennox, J., Westerlund, M., Wu, W., and C. Perkins, "Sending Multiple Media Streams in a Single RTP Session: Grouping RTCP Reception Statistics and Other Feedback", draft-ietf-avtcore-rtp-multi-stream-optimisation-04 (work in progress), August 2014.

[I-D.ietf-mmusic-sdp-bundle-negotiation]
Holmberg, C., Alvestrand, H., and C. Jennings, "Negotiating Media Multiplexing Using the Session Description Protocol (SDP)", draft-ietf-mmusic-sdp-bundle-negotiation-08 (work in progress), August 2014.

[I-D.ietf-rmcat-cc-requirements]
Jesup, R., "Congestion Control Requirements For RMCAT", draft-ietf-rmcat-cc-requirements-05 (work in progress), July 2014.

[I-D.ietf-rtcweb-data-channel]
Jesup, R., Loreto, S., and M. Tuexen, "WebRTC Data Channels", draft-ietf-rtcweb-data-channel-11 (work in progress), July 2014.

[I-D.ietf-rtcweb-overview]
Alvestrand, H., "Overview: Real Time Protocols for Browser-based Applications", draft-ietf-rtcweb-overview-11 (work in progress), August 2014.

[I-D.ietf-rtcweb-rtp-usage]
Perkins, C., Westerlund, M., and J. Ott, "Web Real-Time Communication (WebRTC): Media Transport and Use of RTP", draft-ietf-rtcweb-rtp-usage-17 (work in progress), August 2014.

[I-D.ietf-rtcweb-transports]
Alvestrand, H., "Transports for WebRTC", draft-ietf-rtcweb-transports-06 (work in progress), August 2014.

[I-D.petithuguenin-avtcore-rfc5764-mux-fixes]
Petit-Huguenin, M. and G. Salgueiro, "Multiplexing Scheme Updates for Secure Real-time Transport Protocol (SRTP) Extension for Datagram Transport Layer Security (DTLS)", draft-petithuguenin-avtcore-rfc5764-mux-fixes-00 (work in progress), July 2014.

[I-D.welzl-rmcat-coupled-cc]
Welzl, M., Islam, S., and S. Gjessing, "Coupled congestion control for RTP media", draft-welzl-rmcat-coupled-cc-03 (work in progress), May 2014.

[RFC2697] Heinanen, J. and R. Guerin, "A Single Rate Three Color Marker", RFC 2697, September 1999.

[RFC2698] Heinanen, J. and R. Guerin, "A Two Rate Three Color Marker", RFC 2698, September 1999.

[RFC2914] Floyd, S., "Congestion Control Principles", BCP 41, RFC 2914, September 2000.

[RFC3168] Ramakrishnan, K., Floyd, S., and D. Black, "The Addition of Explicit Congestion Notification (ECN) to IP", RFC 3168, September 2001.

[RFC3270] Le Faucheur, F., Wu, L., Davie, B., Davari, S., Vaananen, P., Krishnan, R., Cheval, P., and J. Heinanen, "Multi-Protocol Label Switching (MPLS) Support of Differentiated Services", RFC 3270, May 2002.

[RFC4103] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.

[RFC4303] Kent, S., "IP Encapsulating Security Payload (ESP)", RFC 4303, December 2005.

[RFC4566] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[RFC4594] Babiarz, J., Chan, K., and F. Baker, "Configuration Guidelines for DiffServ Service Classes", RFC 4594, August 2006.

[RFC5109] Li, A., "RTP Payload Format for Generic Forward Error Correction", RFC 5109, December 2007.

[RFC5127] Chan, K., Babiarz, J., and F. Baker, "Aggregation of Diffserv Service Classes", RFC 5127, February 2008.

[RFC5129] Davie, B., Briscoe, B., and J. Tay, "Explicit Congestion Marking in MPLS", RFC 5129, January 2008.

[RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5389] Rosenberg, J., Mahy, R., Matthews, P., and D. Wing, "Session Traversal Utilities for NAT (STUN)", RFC 5389, October 2008.

[RFC5462] Andersson, L. and R. Asati, "Multiprotocol Label Switching (MPLS) Label Stack Entry: "EXP" Field Renamed to "Traffic Class" Field", RFC 5462, February 2009.

[RFC5761] Perkins, C. and M. Westerlund, "Multiplexing RTP Data and Control Packets on a Single Port", RFC 5761, April 2010.

[RFC5764] McGrew, D. and E. Rescorla, "Datagram Transport Layer Security (DTLS) Extension to Establish Keys for the Secure Real-time Transport Protocol (SRTP)", RFC 5764, May 2010.

[RFC5766] Mahy, R., Matthews, P., and J. Rosenberg, "Traversal Using Relays around NAT (TURN): Relay Extensions to Session Traversal Utilities for NAT (STUN)", RFC 5766, April 2010.

[RFC6062] Perreault, S. and J. Rosenberg, "Traversal Using Relays around NAT (TURN) Extensions for TCP Allocations", RFC 6062, November 2010.

[RFC6437] Amante, S., Carpenter, B., Jiang, S., and J. Rajahalme, "IPv6 Flow Label Specification", RFC 6437, November 2011.

[RFC6458] Stewart, R., Tuexen, M., Poon, K., Lei, P., and V. Yasevich, "Sockets API Extensions for the Stream Control Transmission Protocol (SCTP)", RFC 6458, December 2011.

[W3C.WD-mediacapture-streams-20130903] Burnett, D., Bergkvist, A., Jennings, C., and A. Narayanan, "Media Capture and Streams", World Wide Web Consortium WD WD-mediacapture-streams-20130903, September 2013, <http://www.w3.org/TR/2013/WD-mediacapture-streams-20130903>.

Appendix A. Change History

[To be removed before RFC publication.]

Changes from draft-york-dart-dscp-rtp-00 to -01:

- Added examples (Section 5)
- Reworked text on RTP session multiplexing, at most one RTP session can be used per UDP 5-tuple.
- Initial terminology alignment with RTP grouping taxonomy draft.
- Added Section 2.5 on real-time communication interaction w/ reordering based on text from Harald Alvestrand.
- Strengthened warnings on loss of differentiation, but indicate that differentiation may still be useful from source to point of loss.
- Added a few sentences on DiffServ and MPLS.
- Added discussion of UDP-encapsulated protocols that are reordering sensitive.
- Added initial security considerations.
- Many editorial changes

Changes from draft-york-dart-dscp-rtp-01 to -02:

- More terminology alignment with RTP grouping taxonomy draft: "RTP packet stream" -> "RTP stream"
- Aligned terminology for less-than-best-effort with RFC 3662 - LE (Lower Effort) PHB and service
- Minor reference updates

Changes from draft-york-dart-dscp-rtp-02 to draft-ietf-dart-dscp-rtp-00:

- Reduce author list and convert to Informational - remove RFC 2119 reference and keywords
- Strengthen TCP and SCTP text.
- Add section 2.6 on drop precedence.

- Remove discussion of multiplexing multiple RTP sessions on a single UDP 5-tuple
- Add discussions of RTCP, STUN/ICE/TURN and coupled congestion control
- Many editorial changes.
- Lots of additional references

Changes from draft-ietf-dart-dscp-rtp-00 to -01:

- Merge the two TCP/SCTP guideline bullets.
- Add DCCP and UDP-Lite material, plus reference to RFC 5405 for UDP (and UDP-Lite) usage guidelines.
- Add "attack surface" security consideration.
- Many editorial changes.
- More references, and moved some references to normative.

Changes from draft-ietf-dart-dscp-rtp-01 to -02:

- Reorganize text for better topic flow and make related edits.

Changes from draft-ietf-dart-dscp-rtp-02 to -03:

- Correct text on treatment of excess EF traffic to indicate that excess traffic is dropped.
- Say that transport protocol design and implementation guidance is not provided on use of multiple DSCPs and PHBs in a single transport protocol connection or association.
- RTCWEB -> WebRTC, and correct problems in descriptions of how it uses multiplexing.
- Fix DCCP congestion control discussion and text on coupled congestion controllers.
- Strengthen text on what happens when TURN selects TCP for NAT traversal.
- Note open issue on how to mark RTCP traffic.
- Many editorial changes.

Changes from draft-ietf-dart-dscp-rtp-03 to -04:

- Add abstract/intro text on SDP bundle usage, e.g., by WebRTC
- Remove erroneous use of SSRC w/source stream in Section 2.1
- Add text on WebRTC data channel examples
- Add text on transport protocol complexities that would be necessary to deal with multiple QoS levels in same protocol connection or association
- Additional minor edits.

Changes from draft-ietf-dart-dscp-rtp-04 to -05:

- Rewrite RTCP text and guidelines, including new section 5.4.
- Use "SSRC" as term for sender of RTP stream.
- Additional minor edits.

Authors' Addresses

David Black (editor)
EMC
176 South Street
Hopkinton, MA 01748
USA

Phone: +1 508 293-7953
Email: david.black@emc.com

Paul Jones
Cisco
7025 Kit Creek Road
Research Triangle Park, MA 27502
USA

Phone: +1 919 476 2048
Email: paulej@packetizer.com