

DSCP and other packet markings for RTCWeb QoS  
draft-ietf-rtcweb-qos-00

Abstract

Many networks, such as Service Provider and Enterprise networks, can provide per packet treatments based on Differentiated Services Code Points (DSCP) on a per hop basis. This document defines the recommended DSCP values for browsers to use for various classes of traffic.

This draft is a very early and far from done. It is meant to provide the structure for the idea of how to do this but much discussion is needed about the details.

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 April 15, 2013.

Copyright Notice

Copyright (c) 2012 IETF Trust and the persons identified as the document authors. All rights reserved.

This document is subject to [BCP 78](#) and the IETF Trust's Legal Provisions Relating to IETF Documents (<http://trustee.ietf.org/license-info>) in effect on the date of publication of this document. Please review these documents carefully, as they describe your rights and restrictions with respect to this document. Code Components extracted from this document must include Simplified BSD License text as described in [Section 4.e](#) of the Trust Legal Provisions and are provided without warranty as described in the Simplified BSD License.

## Table of Contents

1. Introduction . . . . .	3
2. Relation to Other Standards . . . . .	3
3. Terminology . . . . .	4
4. Inputs . . . . .	4
5. DSCP Mappings . . . . .	4
6. QCI Mapping . . . . .	5
7. WiFi Mapping . . . . .	5
8. W3C API Implications . . . . .	6
9. Security Considerations . . . . .	6
10. IANA Considerations . . . . .	6
11. Acknowledgements . . . . .	6
12. Appendix: Code Hints . . . . .	7
13. Normative References . . . . .	9
Authors' Addresses . . . . .	9

## 1. Introduction

DiffServ style packet marking can help provide QoS in some environments. There are many use cases where such marking does not help, but it seldom makes things worse, if packets are marked appropriately. In other words, when attempting to avoid congestion by marking certain traffic flows, say all audio or all audio and video, marking too many audio and/or video flows for a given network's capacity can prevent desirable results. Either too much other traffic will be starved, or there is not enough capacity for the preferentially marked packets (i.e., audio and/or video).

This draft proposes how a browser and other VoIP applications can mark packets. This draft does not contradict or redefine any advice from previous IETF RFCs but simply provides a simple set of recommendations for implementors based on the previous RFCs.

There are some environments where priority markings frequently help. These include:

1. If the congested link is the broadband uplink in a Cable or DSL scenario, often residential routers/NAT support preferential treatment based on DSCP.
2. If the congested link is a local WiFi network, marking may help.
3. In some cellular style deployments, markings may help in cases where the network does not remove them.

Traditionally DSCP values have been thought of as being site specific, with each site selecting its own code points for each QoS level. However in the RTCWeb use cases, the browsers need to set them to something when there is no site specific information. This document describes a reasonable default set of DSCP code point values drawn from existing RFCs and common usage. These code points are solely defaults. Future drafts may define mechanisms for site specific mappings to override the values provided in this draft.

This draft defines some inputs that the browser can look at to determine how to set the various packet markings and defines the a mapping from abstract QoS policies (media type, priority level) to those packet markings.

## 2. Relation to Other Standards

This specification does not change or override the advice in any other standards about setting packet markings. It simply provides

a non-normative summary of them and provides the context of how they relate into the RTCWeb context. It also specified the requirements for the W3C API to understand what it needs to control and how the control splits between things the JavaScript application running in the browser can control and things the browser needs to control. In some cases, such as DSCP where the normative RFC leaves open multiple options to choose from, this clarifies which choice should be used in the RTCWeb context.

### 3. Terminology

TODO - add the boiler plate

### 4. Inputs

The first input is the type of the media. The browser provides this input as it knows if the media is audio, video, or data. In this specification, both interactive and streaming media is included. They are treated in different categories as their QoS requirements are slightly different. The second input is the relative treatment of the stream within that session. Many applications have multiple video streams and often some are more important than others. JavaScript applications can tell the browser whether a particular media stream is high, medium, or low importance to the application.

### 5. DSCP Mappings

Below is a table of DSCP markings for each application type RTCWeb is interested in. These DSCPs for each application type listed are a reasonable default set of code point values, and currently not mandatory for every usage. For example, some networks may have a policy in place to have Interactive Video use the EF DSCP.

Application Type	Low	Medium	High
Audio	46 (EF)	46 (EF)	46 (EF)
Interactive Video	38 (AF43)	36 (AF42)	34 (AF41)
Non-Interactive Video	26 (AF33)	28 (AF32)	30 (AF31)
Data	8 (CS1)	0 (BE)	10 (AF11)

Table 1

[Editor's Note: the application type is currently inconsistent with similar applications defined in [6]. Further discussion is likely needed to resolve this.]

## 6. QCI Mapping

Application Type	Low	Medium	High
Audio	1	1	1
Interactive Video	2	2	2
Non-Interactive Video	8	6	4
Data	9	9	3

Table 2

This corresponds to the mapping provided in TODO REF which are: QCI values (LTE)

Value			Use
1	GBR	2	Interactive Voice
2	GBR	4	Interactive Video
3	GBR	5	Non-Interactive Video
4	GBR	3	Real Time Gaming
5	Non-BG	R 1	IMS Signalling
6	Non-BG	R 7	interactive Voice, video, games
7-9	Non-BG	R 6	non interactive video / TCP web, email, / Platinum vs gold user

Table 3

## 7. WiFi Mapping

	Low	Medium	High
Audio	6	6	6
Interactive Video	5	5	5
Non-Interactive Video	4	4	4
Data	1	0	3

Table 4

This corresponds to the mappings from TODO REF of

Value		Traffic Type	Access Category (AC)	Designation
1	BK	Background	AC_BK	Background
2	-	(spare)	AC_BK	Background
0	BE	Best Effort	AC_BE	Best Effort
3	EE	Excellent Effort	AC_BE	Best Effort
4	CL	Controlled Load	AC_VI	Video
5	VI	Video	AC_VI	Video
6	VO	Voice	AC_VO	Voice
7	NC	Network Control	AC_VO	Voice

Table 5

## 8. W3C API Implications

To work with this proposal, the W3C specification would need to provide a way to specify the importance of media and data streams.

The W3C API should also provide a way for the application to find out the source and destination IP and ports of any flow as well as the DSCP value or other markings in use for that flow. The JavaScript application can then communicate this to a web service that may install a particular policy for that flow.

[Editor's Note: the idea of bundling applications/media needs to be further explored.]

## 9. Security Considerations

TODO - discuss implications of what browser can set and what JavaScript can set

## 10. IANA Considerations

This specification does not require any actions from IANA.

## 11. Acknowledgements

Cullen Jennings was one of the authors of this text in the original individual submission but was unceremoniously kicked off by the chairs when it became a WG version. Thanks for hints on code to do this from Paolo Severini, Jim Hasselbrook, Joe Marcus, and Erik Nordmark.

## 12. Appendix: Code Hints

On windows setting the source interface works but BSD, OSX, Linux use weak end-system model and will route out different interface if that looks like a better route. (TODO - Can someone verify this with specific versions?)

In windows you might be able to tell something about priority of an interface for ICE purposes with WlanQueryInterface or GetIfTable.

The specific mechanisms required to set DSCP code points depend on the application platform.

In windows, setting the DSCP is not easy. See Knowledge Base Article KB248611. TODO - add more information about what can be done for windows.

For most unix variants, the following program can set DSCP.

TODO - make this work in V6. For v6 have a look at IPv6\_TCLASS or better the tclass part of sin6\_flowid for IPv6

TODO - Can someone test and report back results of program in iOS, Android, Linux, OSX, BSD.

Example test program:

```
#include <sys/types.h>
#include <sys/socket.h>
#include <netdb.h>
#include <netinet/in.h>
#include <arpa/inet.h>
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <errno.h>
#include <unistd.h>

#define MSG "Hello, World!"

int
main(void) {
    int sock = -1;
    struct sockaddr *local_addr = NULL;
    struct sockaddr_in sockin, host;
    int tos = 0x60; /* CS3 */
```

```
socklen_t socksiz = 0;
char *buffer = NULL;

sock = socket(AF_INET, SOCK_DGRAM, 0);
if (sock < 0) {
    fprintf(stderr, "Error: %s\n", strerror(errno));
    exit(-1);
}

memset(&sockin, 0, sizeof(sockin));
sockin.sin_family = PF_INET;
sockin.sin_addr.s_addr = inet_addr("11.1.1.1");
socksiz = sizeof(sockin);

local_addr = (struct sockaddr *) &sockin;

/* Set ToS/DSCP */
if (setsockopt(sock, IPPROTO_IP, IP_TOS, &tos,
               sizeof(tos)) < 0) {
    fprintf(stderr, "Error setting TOS: %s\n", strerror(errno));
}

/* Bind to a specific local address */
if (bind(sock, local_addr, socksiz) < 0) {
    fprintf(stderr, "Error binding to socket: %s\n", strerror(errno));
    close(sock); sock=-1;
    exit(-1);
}

buffer = (char *) malloc(strlen(MSG) + 1);
if (buffer == NULL) {
    fprintf(stderr, "Error allocating memory: %s\n", strerror(errno));
    close(sock); sock=-1;
    exit(-1);
}
strcpy(buffer, MSG, strlen(MSG) + 1);
memset(&host, 0, sizeof(host));
host.sin_family = PF_INET;
host.sin_addr.s_addr = inet_addr("10.1.1.1");
host.sin_port = htons(12345);

if (sendto(sock, buffer, strlen(buffer), 0,
           (struct sockaddr *) &host, sizeof(host)) < 0) {
    fprintf(stderr, "Error sending message: %s\n", strerror(errno));
    close(sock); sock=-1;
    free(buffer); buffer=NULL;
    exit(-1);
}
```



```
    free(buffer); buffer=NULL;
    close(sock); sock=-1;

    return 0;
}
```

### 13. Normative References

- [1] Babiarz, J., Chan, K., and F. Baker, "Configuration Guidelines for DiffServ Service Classes", [RFC 4594](#), August 2006.
- [2] 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.
- [3] 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.
- [4] Heinanen, J., Baker, F., Weiss, W., and J. Wroclawski, "Assured Forwarding PHB Group", [RFC 2597](#), June 1999.
- [5] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [6] Polk, J., Dhesikan, S., "The Session Description Protocol (SDP) 'trafficclass' Attribute", "work in progress", July 2012

### Authors' Addresses

Subha Dhesikan  
Cisco

Email: [sdhesika@cisco.com](mailto:sdhesika@cisco.com)

Dan Druta (editor)  
ATT

Email: [dd5826@att.com](mailto:dd5826@att.com)

Paul Jones  
Cisco

Email: [paulej@packetizer.com](mailto:paulej@packetizer.com)

James Polk  
Cisco

Email: [jmpolk@cisco.com](mailto:jmpolk@cisco.com)