RTC-Web Non-Media Data Transport Requirements
draft-cbran-rtcweb-data-00

## Abstract

This document outlines a protocol and requirements for RTC-Web client
application to transmit real-time, non-media data.

## Status of this Memo

## Copyright Notice

## Table of Contents

## [1.](#) Introduction

This specification will focus on the transport of real-time non-media data requirements for RTC-Web client applications. An example of real-time non-media data, would be a characters screen position within an multiplayer HTML5 video game.
The non-media data transport requirements fit into a series of specifications have been created to address RTC-Web negotiation and signaling protocols, security requirements, media transmission and use cases. More information on the RTC-Web can be found here:
[TODO put links to supporting drafts here]

## [2.](#) Terminology

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" in this document are to be interpreted as described in [RFC 2119](#) [RFC2119].

## [3.](#) Non-Media Data Transport Requirements

The RTC-WEB will enable for rich voice and video communications from client applications, such as a web browser. One of the natural extensions of the RTC-WEB and the work emerging from the HTML5 community is video games. Video games have a similar stringent real-time requirement for exchanging non-media data types such as a player's screen position.
The question of how best to handle non-media data types has been raised. There have been proposals to address this problem. Common to all proposals is how the data transport session is set up, using ICE [RFC5245] in a similar manner to that of RTP [RFC3550]. The proposals vary from once the session is set up; one proposal is just to use a thin shim on top of UDP or DTLS to de-multiplex the packets from other packets such as RTP on the same connection. Another proposal is DTLS over DCCP over UDP with some appropriate congestion control scheme chosen for DCCP. Lastly there has been a proposal to define a data codec to carry the data in RTP.
Of all the solutions proposed to date there have been issues with implementation maturity and availability, congestion control, high overhead and NAT traversal. The solution outlined within this

specification proposes a lightweight, simple to implement approach for RTC-Web client applications to transmit real-time non-media data.

## 4. Solution Overview

Each application datagram is send with a single byte header to help with de-multiplexing issues. The combined datagraph and header are sent either over UDP or DTLS to the receiver. The receiver sends an acknowledgment for every packet it receives. The sender computes a packet loss rate based upon the number of packets sent, and number of acknowledgements it has received inside of given time window. The browser, or other RTC-Web client application implementation, then enforces a maximum bandwidth usage using the TFRC-SP[RFC4828]. Practically this can be implemented with a simple lookup table such as Table 1 in [RFC4828] that limits the data rate. A JavaScript application running in the browser can implement more complex fragmentation, reliability, and congestion control algorithms, but it is the browser, or other RTC-Web client application, that is responsible for enforcing the basic congestion safety with the TFRC-SP algorithm.

## 5. Specification

When sending a datagram, a single byte with the value 62 MUST be prepended to the data to be sent. The data is then sent over the UDP or a DTLS flow that has been set up by ICE. The receiver MUST send an acknowledgement for each packets it receives. This acknowledgment is a UDP or DTLS datagram containing a single byte with the value of 63. The packet loss rate is computed by comparing the number of packet sent to the of acknowledgements received within the past 2 seconds. The packet loss rate and amount of data sent is used with the TFRC-SP algorithm to compute a maximum safe bandwidth. The sender MUST not exceed this bandwidth. If an application requests the browser, or other RTC-Web client application, to send a packet that would exceed the bandwidth, the RTC-Web client application MUST signal an error to the requesting application and drop the packet.

## 6. IANA Considerations

This document makes no request of IANA.
Note to RFC Editor: this section may be removed on publication as an RFC.

## 7. Security Considerations

Because there are a number of security issues, considerations and requirements for RTC-WEB client applications there is a draft that specifically addresses the RTC-WEB application security considerations. This draft defers it's security considerations and requirements to the security considerations for RTC-Web draft [I-D.ekr-security-considerations-for-rtc-web].

## 8. Acknowledgements

You too can see your name here, just send us comments.

## 9. References

| [RFC2119] | Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", BCP 14, RFC 2119, March 1997. |
|---|---|
| [RFC3550] | Schulzrinne, H., Casner, S., Frederick, R. and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", STD 64, RFC 3550, July 2003. |
| [RFC4828] | Floyd, S. and E. Kohler, "TCP Friendly Rate Control (TFRC): The Small-Packet (SP) Variant", RFC 4828, April 2007. |
| [RFC5245] | Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010. |
| [I-D.ekr-security-considerations-for-rtc-web] | Rescorla, E.K., "Security Considerations for RTC-Web", May 2011. |

## Authors' Addresses

Cary Bran Bran Cisco 170 West Tasman Drive San Jose, CA 95134 USA Phone: +1 206 256-3502 EMail: cbran@cisco.com

Cullen Jennings Jennings Cisco 170 West Tasman Drive San Jose, CA 95134 USA Phone: +1 408 421-9990 EMail: fluffy@cisco.com