Network Working Group	H. Alvestrand
Internet-Draft	Google
Intended status: Experimental Protocol	February 15, 2011
Expires: August 19, 2011	

A Datagram Transport for the RTC-Web profile
draft-alvestrand-dispatch-rtcweb-datagram-01

Abstract

This document describes a combination and profiling of existing IETF protocols to provide a datagram service that is suitable as a generic transport substrate for the RTC-Web family of real-time audio/video applications.

Requirements Language

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]*.

Status of this Memo

Copyright Notice

Table of Contents

1. Introduction

When transporting audio / video and other realtime data between participants on the current Internet, there are a number of obstacles to be faced.

Among them are NAT boxes, firewalls, connection interruptions, the availability of multiple paths between participants, and capacity issues.
This memo describes a combination of existing protocols that can be used to achieve a seamless datagram transport service across this very heterogenous environment.
An overview of the effort of which this is a part can be found in the overview document, [overview].

2. Terminology

This draft uses a couple of commonly used terms in quite specific ways. The reader is advised to study these definitions carefully.
(TODO: Agree on terminology to use)

Session An association with two endpoints, between which datagrams flow.

Datagram A sequence of octets, of a given length. In this specification, a datagram does not carry addressing information.

Channel One means of transporting a datagram over a session. A session may have multiple channels at any time. <Question: Should this word be replaced by "transport", for more consistency with ICE?>

Endpoint One end of a session. This document does not distinguish between an initiator and a responder endpoint.

Control channel A means of communication between the endpoints of a session that does not require a transport to be active. Typically, authentication, authorization and negotiation is carried out over the control channel. The specification of the control channel is out of scope for this specification.

3. Service model

The basic model presented is a datagram model. On top of this one can layer various services, such as pseudoTCP (REF), RTP[RFC3550] or any other higher layer protocol that is capable of running across a datagram service. (If a TCP connection can be established between the parties, this is usually the preferred option for reliable, sequenced transfer. The use of this datagram service for reliable transfer should be considered an option available for the case where only UDP connectivity is available.)
The addressing model departs from the traditional Internet model in that end point addresses are not used for endpoint identification, only for channel establisment; instead, an initial packet exchange, using ICE [RFC5245], is used to bind a channel to a prenegotiated session.
The datagram service is not completely transparent; in particular, it is not possible to carry a datagram where the two highest bits of the

first octet are zero and octet 5 to 8 contain the value 0x2112A442, since these datagrams are reserved for use of the STUN protocol (RFC 5389 section 6).

4. Channel types

4.1. UDP channel

An UDP channel is negotiated using ICE. Each datagram is simply carried as the content of an UDP packet.

4.2. TCP channel

A TCP channel consists of a TCP connection, over which are sent datagrams packaged according to RFC 4571 [RFC4571]. The binding of a TCP channel is done by executing an ICE negotiation over the first few packets passed across the TCP channel, as specified in ICE-TCP [I-D.ietf-mmusic-ice-tcp]

4.3. TLS channel

A TLS channel consists of a standard TLS negotiation, followed by passing datagrams over the TLS record layer[RFC5246] section 6.2. There is no extra length field. A TLS channel is bound to its session by <insert process description>.

4.4. DTLS channel

A DTLS channel is created by executing a DTLS [RFC5238] connection negotiation, followed by datagram exchange, where the datagrams are protected by DTLS mechanisms. The DTLS channel is bound to its session by <insert process>.

4.5. WebSockets channel

A WebSockets channel uses the WebSockets protocol[I-D.ietf-hybi-thewebsocketprotocol] to pass datagrams as binary packets.

4.6. Channels with relay

If there is no possibility of setting up a direct connection, a relay must be used. When both parties are reachable using UDP candidates, the specification from TURN [RFC5766]is used. <NOTE - more text needed here - in particular because for TLS and WebSockets channels, TURN does not apply.>

5. Channel setup, teardown and usage

The service model envisioned here is that all datagrams arriving on a session are considered equally valid. The session gives no guarantees

against duplication, loss or reordering; such concerns are left to the higher protocol layers.
The expected normal usage is that two endpoints will exchange addressing information that can be used for a series of potential channels, that the endpoints will probe for working channels using ICE (RFC 5245), and use the "best" candidate, while using the STUN probing facilities to keep some number of "second best" candidates alive if the "best" candidate stops working.
A data-sending endpoint may unilaterally decide to start or stop using an established channel at any time. No negotiation is necessary.
A receiving endpoint will learn that a channel has been removed by not seeing any more STUN keepalive messages on that channel within <timeout>.
A session is considered closed when all channels that have been successfully established have timed out.

6. An URI scheme for datagram channels

This URI scheme is mainly included in order to make it easy for APIs that normally use URIs as what they use to refer to objects. It reflects exactly the information found in the SDP attributes defined in RFC 5245 section 15.
<NOTE IN DRAFT: This may be replaced with a JSON representation, an XML representation or the SDP representation in later drafts, if the WG so decides.>
The DGSESSION URI scheme specifies the information required for a session; it consists of two parts:

- An absolute reference, which includes the user name and password used to establish channels over this connection.

- A series of addressing hints, which include the data necessary to establish a channel.

<TODO: Fill out an URI registration template for the scheme>
Example:
dgsession:username:password?ipv4:12.34.56:udp:
12345&ipv6:2002::dead:beef:tcp:80&ipv4:12.34.56.78:tls:443
The sequence of addressing hints is an indication of the preference of the URL constructor for the sequence in which to try these candidates; the most preferred address is the one to the left.
Note that a DGSESSION URI is a capability; anyone with the URI will be able to connect to the entity. They should therefore be handled in the same way as (short-term) passwords, and never passed in the clear.

6.1. new section

7. IANA Considerations

This document registers the URI scheme from section Section 7, Paragraph 1.
Note to RFC Editor: this section may be removed on publication as an RFC.

8. Security Considerations

As with all layered protocols, it is a matter for the application to decide which level security should be provided at. For instance, an RTP session protected using SRTP <ref> can be considered to not need any further safeguards against interception, modification or replay, so can be passed "in the clear" across any channel type here. For data without such protection, adequate measures need to be taken; in particular, it is trivially easy for someone with the ability to snoop and insert packets to insert fake packets into an established UDP channel.
The main defense against denial-of-service attacks is the fact that the ICE mechanisms were designed for low cost refusal of unauthorized connections.

9. Acknowledgements

Thanks to Markus Isomaki for reviewing version -00.

10. References

10.1. Normative 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.
[RFC4571]	Lazzaro, J., "Framing Real-time Transport Protocol (RTP) and RTP Control Protocol (RTCP) Packets over Connection-Oriented Transport", RFC 4571, July 2006.
[RFC5238]	Phelan, T., "Datagram Transport Layer Security (DTLS) over the Datagram Congestion Control Protocol (DCCP)", RFC 5238, May 2008.
[RFC5245]	Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", RFC 5245, April 2010.

[RFC5246]	Dierks, T. and E. Rescorla, "The Transport Layer Security (TLS) Protocol Version 1.2", RFC 5246, August 2008.
[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.
[I-D.ietf-mmusic-ice-tcp]	Rosenberg, J, Keranen, A, Lowekamp, B and A Roach, "TCP Candidates with Interactive Connectivity Establishment (ICE)", Internet-Draft draft-ietf-mmusic-ice-tcp-16, November 2011.
[I-D.ietf-hybi-thewebsocketprotocol]	Fette, I and A Melnikov, "The WebSocket protocol", Internet-Draft draft-ietf-hybi-thewebsocketprotocol-17, September 2011.

10.2. Informative References

[overview]	Alvestrand, H, "Overview: Real Time Protocols for Brower-based Applications", November 2010.

Appendix A. Change history

Appendix A.1. Changes from alvestrand-00 to alvestrand-01

Added the WebSockets channel option. Made some changes and clarifications, mainly based on Markus Isomaki's review. Pointed out that the DGSESSION URI scheme has to represent exactly the semantics of the SDP extensions for ICE.

Author's Address

Harald Tveit Alvestrand Alvestrand Google Kungsbron 2
Stockholm, 11122 Sweden EMail: harald@alvestrand.no