

RTCWEB  
Internet-Draft  
Intended status: Standards Track  
Expires: February 13, 2015

M. Perumal  
Ericsson  
D. Wing  
R. Ravindranath  
T. Reddy  
Cisco Systems  
M. Thomson  
Mozilla  
August 12, 2014

**STUN Usage for Consent Freshness**  
**draft-ietf-rtcweb-stun-consent-freshness-06**

**Abstract**

To prevent sending excessive traffic to an endpoint, periodic consent needs to be obtained from that remote endpoint.

This document describes a consent mechanism using a new STUN usage. This same mechanism can also determine connection loss ("liveness") with a remote peer.

**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 February 13, 2015.

**Copyright Notice**

Copyright (c) 2014 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

<a href="#">1.</a>	Introduction . . . . .	<a href="#">2</a>
<a href="#">2.</a>	Terminology . . . . .	<a href="#">3</a>
<a href="#">3.</a>	Design Considerations . . . . .	<a href="#">3</a>
<a href="#">4.</a>	Solution . . . . .	<a href="#">3</a>
<a href="#">4.1.</a>	Expiration of Consent . . . . .	<a href="#">4</a>
<a href="#">4.2.</a>	Immediate Revocation of Consent . . . . .	<a href="#">5</a>
<a href="#">5.</a>	Connection Liveness . . . . .	<a href="#">5</a>
<a href="#">6.</a>	DiffServ Treatment for Consent packets . . . . .	<a href="#">6</a>
<a href="#">7.</a>	W3C API Implications . . . . .	<a href="#">6</a>
<a href="#">8.</a>	Security Considerations . . . . .	<a href="#">6</a>
<a href="#">9.</a>	IANA Considerations . . . . .	<a href="#">7</a>
<a href="#">10.</a>	Acknowledgement . . . . .	<a href="#">7</a>
<a href="#">11.</a>	References . . . . .	<a href="#">7</a>
<a href="#">11.1.</a>	Normative References . . . . .	<a href="#">7</a>
<a href="#">11.2.</a>	Informative References . . . . .	<a href="#">7</a>
	Authors' Addresses . . . . .	<a href="#">8</a>

## [1.](#) Introduction

To prevent attacks on peers, RTP endpoints have to ensure the remote peer wants to receive traffic. This is performed both when the session is first established to the remote peer using ICE connectivity checks, and periodically for the duration of the session using the procedures defined in this document.

When a session is first established, WebRTC implementations are required to perform STUN connectivity checks as part of ICE [[RFC5245](#)]. That initial consent is not described further in this document and it is assumed that ICE is being used for that initial consent.

Related to consent is loss of connectivity ("liveness"). Many applications want notification of connection loss to take appropriate actions (e.g., alert the user, try switching to a different interface).

This document describes a new STUN usage with exchange of request and response messages to verify the remote peer's consent to receive



traffic, and the absence of which for a period of time indicates a loss of liveness.

WebRTC endpoints are required to support full ICE as specified in section 3.4 of [[I-D.ietf-rtcweb-transports](#)]. However, when WebRTC endpoints interwork with other endpoints that support only ICE-lite (e.g., gateways) those endpoints will not generate consent checks, but just respond to consent checks they receive.

## **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 [[RFC2119](#)].

**Consent:** It is the mechanism of obtaining permission to send traffic to a certain transport address. This is the initial consent to send traffic, which is obtained by ICE or a TCP handshake.

**Consent Freshness:** Permission to continue sending traffic to a certain transport address. This is performed by the procedure described in this document.

**Session Liveness:** Detecting loss of connectivity to a certain transport address. This is performed by the procedure described in this document.

**Transport Address:** The remote peer's IP address and (UDP or TCP) port number.

## **3. Design Considerations**

Although ICE requires periodic keepalive traffic to keep NAT bindings alive ([Section 10 of \[RFC5245\]](#), [[RFC6263](#)]), those keepalives are sent as STUN Indications which are send-and-forget, and do not evoke a response. A response is necessary both for consent to continue sending traffic, as well as to verify session liveness. Thus, we need a request/response mechanism for consent freshness. ICE can be used for that mechanism because ICE already requires ICE agents continue listening for ICE messages, as described in [section 10 of \[RFC5245\]](#).

## **4. Solution**

There are two ways consent to send traffic is revoked: expiration of consent and immediate revocation of consent, which are discussed in the following sections.



#### **4.1. Expiration of Consent**

A WebRTC browser performs a combined consent freshness and session liveness test using STUN request/response as described below:

An endpoint **MUST NOT** send application data (e.g., RTP, RTCP, SCTP, DTLS), over any transport protocol (e.g., UDP, TCP) on an ICE-initiated connection unless the receiving endpoint consents to receive the data. After a successful ICE connectivity check on a particular transport address, subsequent consent **MUST** be obtained following the procedure described in this document. The consent expires after a fixed amount of time. During ICE restart consent checks **MUST** continue to be sent on previously validated pair, and **MUST** be responded to on the previously validated pair, until ICE restart completes.

Note: Although TCP has its own consent mechanism (TCP acknowledgements), consent is necessary over a TCP connection because it could be translated to a UDP connection (e.g., [[RFC6062](#)]).

Explicit consent to send is obtained by sending an ICE binding request to the remote peer's Transport Address and receiving a matching, authenticated, non-error ICE binding response from the remote peer's Transport Address. These ICE binding requests and responses are authenticated using the same short-term credentials as the initial ICE exchange. Implementations **MUST** cease sending data if their consent expires. To prevent expiry of consent, a STUN binding request **MUST** be sent every N milliseconds, where N is chosen randomly with each consent check in the interval  $[.8N, 1.2N]$  (to prevent network synchronization), where N **SHOULD** be 5000. Using the value 5000 milliseconds and that 20% randomization range, N would be a value between 4000 and 6000. These STUN binding requests for consent are not re-transmitted. Each STUN binding request for consent recalculates a new random value N and a new cryptographically-random [[RFC4086](#)] STUN transaction ID.

The initial Consent to send traffic is obtained by ICE. Consent expires after 30 seconds. That is, if a valid STUN binding response corresponding to one of the STUN requests sent in the last 30 seconds has not been received from the remote peer's Transport Address, the endpoint **MUST** cease transmission on that 5-tuple.

To meet the security needs of consent, an untrusted application (e.g., JavaScript) **MUST NOT** be able to obtain or control the STUN transaction ID, because that enables spoofing STUN responses, falsifying consent.



While TCP affords some protection from off-path attackers ([[RFC5961](#)], [[RFC4953](#)]), there is still a risk an attacker could cause a TCP sender to send packets forever by spoofing ACKs. To prevent such an attack, consent checks MUST be performed over all WebRTC-initiated transport connections, including TCP. In this way, an off-path attacker spoofing TCP segments can not cause a TCP sender to send packets longer than the consent timer (30 seconds).

An endpoint that is not sending any application traffic does not need to obtain consent which can slightly conserve its resources. However, the endpoint needs to ensure its NAT or firewall mappings persist which can be done using keepalive or other techniques (see [Section 10 of \[RFC5245\]](#) and see [[RFC6263](#)]). If the endpoint wants to send application traffic, it needs to first obtain consent if its consent has expired.

#### **[4.2.](#) Immediate Revocation of Consent**

The previous section explained how consent expires due to a timeout. In some cases it is useful to signal a connection is terminated, rather than relying on a timeout. This is done by immediately revoking consent.

Consent for sending traffic on the media or data channel is immediately revoked by receipt of an authenticated message that closes the connection (e.g., a TLS fatal alert) or receipt of a valid and authenticated STUN response with error code Forbidden (403). Those consent revocation messages can be lost on the network, so an implementation wanting to immediately revoke consent needs to remember those credentials until consent expiry (30 seconds).

Receipt of an unauthenticated message that closes a connection (e.g., TCP FIN) does not indicate revocation of consent. Thus, an endpoint receiving an unauthenticated end-of-session message SHOULD continue sending media (over connectionless transport) or attempt to re-establish the connection (over connection-oriented transport) until consent expires or it receives an authenticated message revoking consent.

Note that an authenticated SRTCP BYE does not terminate consent; it only indicates the associated SRTP source has quit.

#### **[5.](#) Connection Liveness**

A connection is considered "live" if packets are received from a remote endpoint within an application-dependent period. An application can request a notification when there are no packets received for a certain period (configurable).





Similarly, if packets haven't been received within a certain period, an application can request a consent check (heartbeat) be generated. These two time intervals might be controlled by the same configuration item.

Sending consent checks (heartbeats) at a high rate could allow a malicious application to generate congestion, so applications **MUST NOT** be able to send heartbeats at an average rate of more than 1 per second.

## **6. DiffServ Treatment for Consent packets**

It is RECOMMENDED that STUN consent checks use the same Diffserv Codepoint markings as the ICE connectivity checks described in [section 7.1.2.4 of \[RFC5245\]](#) for a given 5-tuple.

Note: It is possible that different Diffserv Codepoints are used by different media over the same transport address [[I-D.ietf-tsvwg-rtcweb-qos](#)]. Such a case is outside the scope of this document.

## **7. W3C API Implications**

For the consent freshness and liveness test the W3C specification should provide APIs as described below:

1. Ability for the browser to notify the JavaScript that consent freshness has failed for a 5-tuple and the browser has stopped transmitting on that 5-tuple.
2. Ability for the JavaScript to start and stop liveness test and set the liveness test interval.
3. Ability for the browser to notify the JavaScript that a liveness test has failed for a media stream.

## **8. Security Considerations**

This document describes a security mechanism.

The security considerations discussed in [[RFC5245](#)] should also be taken into account.

SRTP is encrypted and authenticated with symmetric keys; that is, both sender and receiver know the keys. With two party sessions, receipt of an authenticated packet from the single remote party is a strong assurance the packet came from that party. However, when a session involves more than two parties, all of whom know each others



keys, any of those parties could have sent (or spoofed) the packet. Such shared key distributions are possible with some MIKEY [[RFC3830](#)] modes, Security Descriptions [[RFC4568](#)], and EKT [[I-D.ietf-avtcore-srtp-ekt](#)]. Thus, in such shared keying distributions, receipt of an authenticated SRTP packet is not sufficient to verify consent.

## **9. IANA Considerations**

This document does not require any action from IANA.

## **10. Acknowledgement**

Thanks to Eric Rescorla, Harald Alvestrand, Bernard Aboba, Magnus Westerlund, Cullen Jennings, Christer Holmberg, Simon Perreault, Paul Kyzivat, Emil Ivov, and Jonathan Lennox for their valuable inputs and comments.

## **11. References**

### **11.1. Normative References**

- [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", [BCP 14](#), [RFC 2119](#), March 1997.
- [RFC4086] Eastlake, D., Schiller, J., and S. Crocker, "Randomness Requirements for Security", [BCP 106](#), [RFC 4086](#), June 2005.
- [RFC5245] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Protocol for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", [RFC 5245](#), April 2010.
- [RFC6263] Marjou, X. and A. Sollaud, "Application Mechanism for Keeping Alive the NAT Mappings Associated with RTP / RTP Control Protocol (RTCP) Flows", [RFC 6263](#), June 2011.

### **11.2. Informative References**

- [I-D.ietf-avtcore-srtp-ekt]  
McGrew, D. and D. Wing, "Encrypted Key Transport for Secure RTP", [draft-ietf-avtcore-srtp-ekt-02](#) (work in progress), February 2014.
- [I-D.ietf-rtcweb-transports]  
Alvestrand, H., "Transports for RTCWEB", [draft-ietf-rtcweb-transports-05](#) (work in progress), June 2014.



[I-D.ietf-tsvwg-rtcweb-qos]

Dhesikan, S., Jennings, C., Druta, D., Jones, P., and J. Polk, "DSCP and other packet markings for RTCWeb QoS", [draft-ietf-tsvwg-rtcweb-qos-02](#) (work in progress), June 2014.

[RFC3830] Arkko, J., Carrara, E., Lindholm, F., Naslund, M., and K. Norrman, "MIKEY: Multimedia Internet KEYing", [RFC 3830](#), August 2004.

[RFC4568] Andreassen, F., Baugher, M., and D. Wing, "Session Description Protocol (SDP) Security Descriptions for Media Streams", [RFC 4568](#), July 2006.

[RFC4953] Touch, J., "Defending TCP Against Spoofing Attacks", [RFC 4953](#), July 2007.

[RFC5961] Ramaiah, A., Stewart, R., and M. Dalal, "Improving TCP's Robustness to Blind In-Window Attacks", [RFC 5961](#), August 2010.

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

#### Authors' Addresses

Muthu Arul Mozhi Perumal  
Ericsson  
Mahadevapura  
Bangalore, Karnataka 560048  
India

Email: [muthu.arul@gmail.com](mailto:muthu.arul@gmail.com)

Dan Wing  
Cisco Systems  
821 Alder Drive  
Milpitas, California 95035  
USA

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



Ram Mohan Ravindranath  
Cisco Systems  
Cessna Business Park  
Sarjapur-Marathahalli Outer Ring Road  
Bangalore, Karnataka 560103  
India

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

Tirumaleswar Reddy  
Cisco Systems  
Cessna Business Park, Varthur Hobli  
Sarjapur Marathalli Outer Ring Road  
Bangalore, Karnataka 560103  
India

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

Martin Thomson  
Mozilla  
Suite 300  
650 Castro Street  
Mountain View, California 94041  
US

Email: [martin.thomson@gmail.com](mailto:martin.thomson@gmail.com)



