Abstract

Verification of peer consent before sending traffic is necessary in WebRTC deployments to ensure that a malicious JavaScript cannot use the browser as a platform for launching attacks. A related problem is session liveness. WebRTC applications may want to detect connection failure and take appropriate actions. This document describes a STUN usage that enables a WebRTC browser to perform the following on a candidate pair ICE is using for a media component after session establishment:

1. Verify the peer consent for continuing to send traffic.
2. Detect connection failure and notify the JavaScript.

This also serves the purpose of refreshing NAT bindings.

Status of this Memo

This Internet-Draft is submitted to IETF 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 August 29, 2013.

Copyright Notice

Copyright (c) 2013 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. Terminology ........................................... 3
3. Definitions ............................................. 3
4. Design Considerations ................................... 4
5. Solution Overview ...................................... 4
6. W3C API Implications ................................... 5
7. Interaction with Keepalives used for Refreshing NAT Bindings ........................................... 5
8. Security Considerations ................................. 6
9. IANA Considerations .................................... 6
10. Acknowledgement ....................................... 6
11. Normative References .................................. 6
Authors’ Addresses ......................................... 6
1. Introduction

WebRTC implementations obtain the peer consent before sending traffic on candidate media transport addresses. This has two parts:

1. Obtaining peer consent for sending traffic at session establishment.
2. Obtaining peer consent for continuing to send traffic after session establishment.

WebRTC implements are required to perform STUN connectivity checks at session establishment as part of ICE procedures [RFC5245]. This takes care of the first part of the consent verification described above.

After session establishment ICE requires STUN Binding indications to be used for refreshing NAT bindings for a candidate pair ICE is using for a media component. Since a STUN Binding indication does not evoke a response, it cannot be used for the second part of the consent verification described above.

A related problem is session liveness. WebRTC applications may want to detect connection failure on candidate media transport addresses after session establishment and take appropriate actions. Again, the STUN Binding indications in ICE sent after session establishment cannot be used for determining session liveness.

This document describes a STUN usage based on STUN request/response that enables a WebRTC browser to perform the following on a candidate pair ICE is using for a media component after session establishment:

1. Verify the peer consent for continuing to send traffic.
2. Detect connection failure and notify the JavaScript.

This also serves the purpose of refreshing NAT bindings.

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].

3. Definitions
Consent: It is the mechanism of obtaining permission to send traffic on a candidate pair.

Consent Freshness: It is the mechanism of obtaining permission to continue sending traffic on a candidate pair ICE is using for a media component after ICE has concluded.

Session Liveness: It is the mechanism of detecting connectivity on a candidate pair ICE is using for a media component after ICE has concluded.

Transport Address: The combination of an IP address and port number (such as a UDP or TCP port number).

4. Design Considerations

As described earlier, STUN indications are not suitable for performing consent freshness. Hence, performing consent freshness requires the use of STUN request/response.

STUN requires the 96 bits transaction ID to be uniformly and randomly chosen from the interval 0 .. 2**96-1, and be cryptographically random. This is deemed sufficient for consent freshness from a security perspective. However, omitting the MESSAGE-INTEGRITY attribute from STUN Binding request/response to avoid the cost of computing SHA1 would break backward compatibility with ICE/ICE-lite agents.

Though ICE specifies STUN Binding indications to be used for keepalives, it requires that an agent be prepared to receive connectivity check as well. If a connectivity check is received, a response is generated, but there is no impact on ICE processing, as described in section 10 of [RFC5245]

While a WebRTC browser could verify whether the peer continues to send SRTCP reports before sending traffic to the peer, the usage of SRTCP together with SDESC [RFC4568] exposes the media keys to the JavaScript and renders SRTCP unsuitable for consent freshness.

The above considerations suggest that STUN Binding request/response is most suitable for performing consent freshness.

5. Solution Overview

The solution uses two timers:
1. A consent timer $T_c$ whose value is determined by the browser.
2. A packet receipt timer $T_r$ whose value is determined by the application.

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

- Starts a consent timer $T_c$ (no less than 15 sec).
- Starts a packet receipt timer $T_r$ (no less than 1 sec); application configurable.
- When either timer expires it starts a STUN transaction.
- When the STUN transaction succeeds, it re-starts both timers.
- When the STUN transaction fails
  * If the transaction was started by timer $T_c$, it stops sending traffic on that candidate pair.
  * Else it notifies the application of the failure and continues.
- It resets timer $T_r$ on receiving any packet from the other side.

While consent freshness serves as a circuit breaker (if there is a failure the WebRTC browser stops sending all traffic on that candidate pair), determining session liveness serves the purpose of notifying the application of connectivity failure so that the application can take appropriate action.

6. 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 a consent freshness transaction has failed for a stream and the browser has stopped transmitting for that stream.
2. Ability for the JavaScript to set the liveness test interval.
3. Ability for the browser to notify the JavaScript that a liveness test has failed for a stream.

7. Interaction with Keepalives used for Refreshing NAT Bindings

An implementation that performs the procedures described in this document has no need to also perform the keepalives described in ICE [RFC5245] or RTP keepalive [RFC6263], as they both force recurring messages to be sent over the UDP port used by RTP. Thus, an implementation that performs the procedures described in this document SHOULD NOT also do the keepalives described in ICE [RFC5245] or RTP keepalives [RFC6263] for the same UDP port.
8. Security Considerations

TBD

9. IANA Considerations

TBD

10. Acknowledgement

Thanks to Eric Rescorla, Harald Alvestrand, Martin Thomson, Bernard Aboba, Cullen Jennings and Simon Perreault for their valuable inputs and comments

11. Normative References


Authors’ Addresses

Muthu Arul Mozhi Perumal
Cisco Systems
Cessna Business Park
Sarjapur-Marathahalli Outer Ring Road
Bangalore, Karnataka  560103
India

Email: mperumal@cisco.com
Dan Wing
Cisco Systems
821 Alder Drive
Milpitas, California 95035
USA

Email: dwing@cisco.com

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

Email: rmohanr@cisco.com

Hadriel Kaplan
Acme Packet

Email: hkaplan@acmepacket.com