BEHAVE Working Group
Internet-Draft
Intended status: Informational
Expires: August 6, 2007

X. Marjou, Ed.
France Telecom

February 2, 2007

Application Mechanism for maintaining alive the Network Address Translator (NAT) mappings associated to RTP flows. draft-marjou-behave-app-rtp-keepalive-01

Status of this Memo

Copyright Notice

Abstract

This document defines a mechanism that enables applications using Real Time Protocol (RTP) to maintain their RTP Network Address Translator (NAT) mappings alive.

Table of Contents

1. Introduction 3
2. Terminology 4
3. Requirements 4
4. List of Alternatives for Performing RTP Keepalive 4
4.1. UDP Packet of 0-byte 5
4.2. RTCP Packets Multiplexed with RTP Packets 5
4.3. STUN Packet 5
4.4. RTP Packet with Comfort Noise Payload 5
4.5. RTP Packet with No-Op Payload 6
4.6. RTP Packet with Incorrect Version Number 6
4.7. RTP Packet with Unknown Payload Type 6
5. Recommended Solution 6
6. Security Considerations 7
7. Acknowledgements 7
8. References 7
Author's Address 8
Intellectual Property and Copyright Statements 9

1. Introduction

[Note: The content of this draft is basically a copy and paste of the current 7.12 section of ICE [5] concerning binding keepalives requirements that apply to a non ICE agent, or that apply to an ICE agent that communicates with a non-ICE agent. It thus makes sense to extract it in a separate document so that non-ICE agents can refer to non-ICE specification.]

Documents [2] and [3] describe NAT behaviors and point-out that two key aspects of NAT are mappings (a.k.a. bindings) and their refreshment. This introduces a derived requirement for applications engaged in a multimedia session involving NAT traversal: they need to generate a minimum of flow activity in order to maintain the NAT mappings alive.

When applied to applications using RTP [4], the RTP media stream packets themselves normally fulfill this requirement. However, as described in ICE [5], there exist some cases where RTP do not generate a minimum flow activity.

The examples are:

- Firstly, in some RTP usages, such as SIP, the media streams can be "put on hold". This is accomplished by using the SDP "sendonly" or "inactive" attributes, as defined in RFC 3264 [6]. RFC 3264 directs implementations to cease transmission of media in these cases. However, doing so may cause NAT bindings to timeout, and media won't be able to come off hold.
- Secondly, some RTP payload formats, such as the payload format for text conversation [7], may send packets so infrequently that the interval exceeds the NAT binding timeouts.
- Thirdly, if silence suppression is in use, long periods of silence may cause media transmission to cease sufficiently long for NAT bindings to time out.

This document first states the requirements that must be supported to perform RTP keepalives (Section 3). In a second step, several alternatives are laid-out to overcome this problem (Section 4). Finally a single solution is recommended, in order to achieve interoperability (Section 5).

The scope of the draft is limited to RTP flows. In particular, this document does not address keepalive activity related to:

- Session signaling flows, such as the Session Initiation Protocol (SIP).
- RTCP flows.

  * Recall that [4] recommends a minimum interval of 5 seconds and that "on hold" procedures of [6] do not impact RTCP transmissions. Therefore, when in use, there is always some RTCP flow activity.
- Other types of flows, such as the Binary Floor Control Protocol (BFCP)

2. Terminology

In this document, the key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "MAY", and "OPTIONAL" are to be interpreted as described in RFC 2119 [1]

3. Requirements

This section outlines the key requirements that the solution need to satisfy in order to provide RTP media keepalive.

REQ 1. The recommended mechanism MUST generate activity within the RTP media stream

REQ 2. The activity is generated periodically for the whole duration of the RTP media stream.

REQ 3. Any type of transport (e.g. UDP, TCP) MUST be supported.

REQ 4. Any type media of stream (e.g. audio, video, text) MUST be supported.

REQ 5. Any type of payload format (e.g. G.711, H.263) MUST be supported.

REQ 6: Session signaling protocols SHOULD not be impacted.

REQ 7: Session description protocols SHOULD not be impacted.

REQ 8: Impacts on existing software SHOULD be minimized.

REQ 9: Remote peer SHOULD not be impacted.

REQ 10: One single mechanism MUST be recommended.

4. List of Alternatives for Performing RTP Keepalive

This section lists some alternatives that could be used in order to

perform a keepalive message within RTP media streams.

A common drawback of most of these alternatives is that they require media packets be sent by the application during "on hold" procedures, which violates the behavior of the inactive and recvonly attributes specified in SDP-NEW [10] and in RFC3264 [6]. Although there can exist some debate whether STUN is a media flow or not, STUN also requires the application to send some packets within the media stream during on-hold procedures.

4.1. UDP Packet of 0-byte

The application sends an empty UDP packet.

Cons:
- This alternative is specific to UDP.
- There may be some implementations that will not ignore these packets.

4.2. RTCP Packets Multiplexed with RTP Packets

The application sends RTCP packets in the RTP media stream itself (i.e. same tuples for both RTP and RTCP packets) [8]. RTCP packets therefore maintain the NAT mappings open.

Cons:
- Multiplexing RTP and RTCP must be supported by the remote peer.
- Multiplexing RTP and RTCP must be signalled in SDP offer/answer.
- This alternative may significantly impact existing software and specifications.

4.3. STUN Packet

The application sends a STUN Binding Request packet and receives a STUN Binding Response [9]

Cons:
- This alternative requires that the remote endpoint support STUN.

4.4. RTP Packet with Comfort Noise Payload

The application sends a RTP packet with a comfort-noise payload [11].

Cons:
- This alternative is limited to voice payload formats only.
- For each payload type, the content of the payload needs to be specified.

4.5. RTP Packet with No-Op Payload

The application sends a RTP No-OP payload [12] .

Cons:
- This payload type needs to be supported by the remote peer.
- This payload type needs to be signalled in SDP offer/answer.

4.6. RTP Packet with Incorrect Version Number

The application sends a RTP with an incorrect version number.

Based on RTP specification [4], the peer should perform a header validity check, and therefore ignore these types of packet.

Cons:
- Only four version numbers are possible. Using one of them for RTP keepalive would be wasteful.

4.7. RTP Packet with Unknown Payload Type

The application sends a RTP packet with an unknown payload type.

Normally the peer will ignore it, as RTP [4] states that "a receiver MUST ignore packets with payload types that it does not understand".

For example, the keepalive RTP packets contain a dynamic payload type that has not been negotiated for the session.

[Note: more details on the selection of the payload type are needed here.]

Cons:
- None

5. Recommended Solution

An application supporting this specification MUST send keepalive packets under the form of ... [Note: The recommended solution needs to be discussed. However recommending a single method among the alternatives of the previous section is the best in term of interoperability. Proposal is the alternative of Section 4.7]

Keepalives packets MUST be sent for each RTP stream regardless of whether the media stream is currently inactive, sendonly, recvonly or sendrecv.

Keepalives packets within a particular RTP media stream MUST use the tuple (source IP address, source TCP/UDP ports, target IP address, target TCP/UDP Port) of the regular RTP packets."

Keepalive packets MUST be sent every Tr seconds. Tr SHOULD be configurable, and otherwise MUST default to 15 seconds. [Note: same value as in [5].]

An application starts sending keepalives packet as soon as the first regular RTP packet of the media stream has been sent. It ceases sending these keepalives packet when the media stream is disabled, or when the communication terminates.

6. Security Considerations

T.B.D.

7. Acknowledgements

Jonathan Rosenberg, via the ICE specification, provided the major inputs for this draft. In addition, thanks to the following folks for useful inputs and comments: Dan Wing, and Aurelien Sollaud.

8. References

[1] Bradner, S., "Key words for use in RFCs to Indicate Requirement Levels", RFC 2119, March 1997.

[2] Audet, F. and C. Jennings, "Network Address Translation (NAT) Behavioral Requirements for Unicast UDP", RFC 4787, January 2007.

[3] Guha, S., Biswas, K., Ford, B., Francis, P., Sivarkumar, S., and P. Srisuresh, "NAT Behavioral Requirements for TCP", draft-ietf-behave-tcp-04 (work in progress), January 2007.

[4] Schulzrinne, H., Casner, S., Frederick, R., and V. Jacobson, "RTP: A Transport Protocol for Real-Time Applications", RFC 3550, July 2003.

[5] Rosenberg, J., "Interactive Connectivity Establishment (ICE): A Methodology for Network Address Translator (NAT) Traversal for Offer/Answer Protocols", draft-ietf-mmusic-ice-13 (work in progress), January 2007.

[6] Rosenberg, J. and H. Schulzrinne, "An Offer/Answer Model with the Session Description Protocol (SDP)", RFC 3264, June 2002.

[7] Hellstrom, G. and P. Jones, "RTP Payload for Text Conversation", RFC 4103, June 2005.

[8] Perkins, C. and M. Magnus, "Multiplexing RTP Data and Control Packets on a Single Port", draft-ietf-avt-rtp-and-rtcp-mux-03 (work in progress), December 2006.

[9] Rosenberg, J., Huitema, C., Mahy, R., and D. Wing, "Simple Traversal Underneath Network Address Translators (NAT) (STUN)", draft-ietf-behave-rfc3489bis-05 (work in progress), October 2006.

[10] Handley, M., Jacobson, V., and C. Perkins, "SDP: Session Description Protocol", RFC 4566, July 2006.

[11] Robert, R., "Real-time Transport Protocol (RTP) Payload for Comfort Noise (CN)", RFC 3389, September 2002.

[12] Andreason, F., Oran, D., and D. Wing, "A No-Op Payload Format for RTP", draft-ietf-avt-rtp-no-op-00 (work in progress), May 2005.

Author's Address

Xavier Marjou (editor)
France Telecom
2, hent Pierre Marzin
Lannion, Brittany 22307
France

Email: xavier.marjou@orange-ftgroup.com

Full Copyright Statement

Intellectual Property

Acknowledgment

Funding for the RFC Editor function is provided by the IETF Administrative Support Activity (IASA).