

Audio/Video Transport
Internet-Draft
Expires: December 21, 2006

J. Lennox
Layered Media
June 19, 2006

Marking and Selectively Retransmitting High-Priority Packets in the
Real-Time Transport Protocol (RTP)
draft-lennox-avt-recoverable-packets-00

Status of this Memo

By submitting this Internet-Draft, each author represents that any applicable patent or other IPR claims of which he or she is aware have been or will be disclosed, and any of which he or she becomes aware will be disclosed, in accordance with [Section 6 of BCP 79](#).

Internet-Drafts are working documents of the Internet Engineering Task Force (IETF), its areas, and its working groups. Note that other groups may also distribute working documents as Internet-Drafts.

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."

The list of current Internet-Drafts can be accessed at
<http://www.ietf.org/ietf/lid-abstracts.txt>.

The list of Internet-Draft Shadow Directories can be accessed at
<http://www.ietf.org/shadow.html>.

This Internet-Draft will expire on December 21, 2006.

Copyright Notice

Copyright (C) The Internet Society (2006).

Abstract

In some circumstances, it is useful and desirable for the sender of an RTP stream to be able to deliver a subset of its RTP packets reliably. One example of this is a video codec which can encode a video stream such that decoder state can be maintained using just a subset of the packets encoded. However, existing RTP reliability mechanisms only define mechanisms which retransmit all the packets of an RTP stream.

This document describes a mechanism by which the sender of an RTP stream can mark a subset of the packets of an RTP stream as high-priority recoverable packets, and by which the receiver of the stream can request retransmission of the high-priority packets.

Table of Contents

1.	Introduction	3
2.	Terminology	3
3.	Overview	4
4.	Use of R Packets With Video Streams	4
5.	RTP Header Extension to Indicate R Packets	5
6.	RTCP Feedback Messages	8
7.	Security Considerations	9
8.	IANA Considerations	10
9.	References	10
9.1.	Normative References	10
9.2.	Informative References	10
Appendix A.	Requirements	11
Appendix B.	Analysis of Alternate Approaches	12
B.1.	Generic NACK	12
B.2.	Generic NACK with Codec knowledge	12
B.3.	Codec-Specific NACK	13
B.4.	Multiple streams	13
	Author's Address	14
	Intellectual Property and Copyright Statements	15

1. Introduction

In many circumstances, it is useful to allow a subset of the packets in a Real-Time Transport Protocol (RTP) [[RFC3550](#)] media stream to be transmitted reliably (i.e. that they be retransmitted if they are lost). However, current mechanisms require either that any lost packet be retransmitted, or that none are. There is no way for the sender of an RTP stream to select some packets of a stream to be retransmittable.

For example, modern flexible video codecs such as H.264 [[ITU.H264.2005](#)] allow video streams to be encoded flexibly. Unlike in earlier video codecs, inter-picture image prediction may be based not just on an immediately preceding picture, but optionally instead on earlier pictures transmitted in the video stream. As a result, it is possible for a video encoder to encode its stream such that in high packet loss situations, stream state can be maintained or re-established using only a fraction of the RTP packets transmitted in the stream. If a media stream experiences transient packet loss, a decoder can avoid reporting all lost packets, and an encoder does not need to retransmit all the missed packets, in order for the decoder to fully re-establish state.

To achieve this desirable property, however, a decoder must be able to determine, based only on the packets it has received, whether a packet it has missed was one that it needed to receive reliably to reestablish stream state. This document describes a payload-format independent mechanism by which a receiver (or intermediate media-aware network entity) can quickly determine whether essential packets have been lost and can request their retransmission.

The mechanism described by this document is also applicable to other application scenarios -- for example, DTMF [[RFC2833](#)] tones in an audio stream -- in which it is useful for some, but not all, of the packets in an RTP stream to be transmitted reliably.

[Appendix A](#) lists the requirements for this protocol, and [Appendix B](#) evaluates some alternate approaches to the problem described.

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](#) [[RFC2119](#)] and indicate requirement levels for compliant implementations.

Lennox

Expires December 21, 2006

[Page 3]

Internet-Draft

Recoverable High-Priority RTP Packets

June 2006

3. Overview

When sending a stream of RTP packets, a media sender may choose to mark a subset of the stream's packets as high-reliability packets. This subset of the RTP packets is named "R packets," for "recoverable" packets. Other packets are called "Non-R packets".

An RTP stream can contain several independent series of R packets. (A given R packet can belong to only one series of R packets). Every R packet is identified with an R packet sequence number (RSeq), unique (modulo 2^{16}) within its series. R packets within a series are numbered in order, and a receiver detects missing R packets by noting gaps in the R packet sequence numbers. R packets sequence numbers are identified using an RTP header extension element, defined in [Section 5](#).

This header extension element is also used in non-R packets, to identify the RTP stream's most recently transmitted R packet in each series. Thus, receivers can detect missing R packets based on non-R packets as well as R packets. Since R packets can be sent at a fairly low rate relative to the overall packet sending rate of the RTP stream, this allows receivers to detect missing R packets much more quickly. If multiple series of R packets are being sent, R packet from one series can also identify the most recently sent R packets from the other active series.

When a receiver detects that it has missed an R packet, it sends an R packet NACK message (RNACK), described in [Section 6](#), using the RTCP feedback (AVPF) [[I-D.ietf-avt-rtcp-feedback](#)] mechanism. On receiving

an RNACK message, the sender can then retransmit the missing packet.

If a sender determines that some R packets, sent in the past, are no longer needed by the receiver, it can send a new R packet indicating that the earlier ones are no longer needed. This R packet is called a superseding R packet, as receipt of a superseding R packet indicates that earlier R packets need not be requested by a receiver. The first R packet of an RTP stream is always encoded such that it supersedes all the previous (non-existent) R packets in the stream.

[4.](#) Use of R Packets With Video Streams

For several video formats, it is possible for a video encoder to encode and packetize its media so that a subset of the packets of an RTP [[RFC3550](#)] stream are sufficient for a stream decoder to fully reestablish the correct decoding state.

In order to take advantage of R packets, an encoder periodically

encodes a frame (or other unit of the media data) in a way that either depends on no previous data (an intra-encoded R frame) or that depends only on previously transmitted R packets (a P-encoded R frame). Frames transmitted in non-R packets can then reference the frames encoded in R packets. If transient packet loss occurs, a decoder need only request retransmission of the R packets it has missed in order to re-establish decoder state; missed non-R packets can be safely ignored. Missed R packets are useful (and necessary) for maintaining the continuity of the media stream. In some cases even if retransmissions of R packets arrive at the decoder too late to be played out, the decoder can "fast-forward" through a subset of the stream in order to catch up to the stream's current playout point.

It is not necessary to receive every R packet ever transmitted on a media stream in order to reestablish the stream state. As mentioned, while some R packets (P frames) depend on previous data transmitted in a stream, others (intra frames) depend on no previous data and imply full stream state. Thus, the R packets carrying intra frames are encoded such that they supersede all the R packets prior to the first packet of the intra frame.

Furthermore, when an encoder receives a report about a lost R packet, it is possible for it to decide to encode a fresh R frame (based only on R frames it believes have been received correctly) rather than retransmitting the lost packet. This frame's R packets are then marked as superseding the R packets of the frames which are no longer used for dependency tracking.

In layered encodings, there can be more than one series of required packets, e.g. a base-layer I-frame eliminates the need for previous base-layer long-term reference frames, but not for enhancement-layer long-term reference frames. In this case, each layer's R frames are identified by a different R packet series.

5. RTP Header Extension to Indicate R Packets

This section describes the RTP header extension element used to indicate R packets.

In an RTP stream containing R packets, packets are marked with an RTP header extension element using the Named RTP Header Extension mechanism [[I-D.ietf-avt-rtp-hdrex](#)].

The R packet header extension element identifies both R packets themselves and previously-sent R packets. This header extension element has the name "org.ietf.avt.r-packet/200606". Every R packet

includes, and every non-R packet SHOULD include, at least one header extension element of this form. It MAY occur multiple times in a header extension if multiple R packet series are in use, but MUST occur only once for any given series.

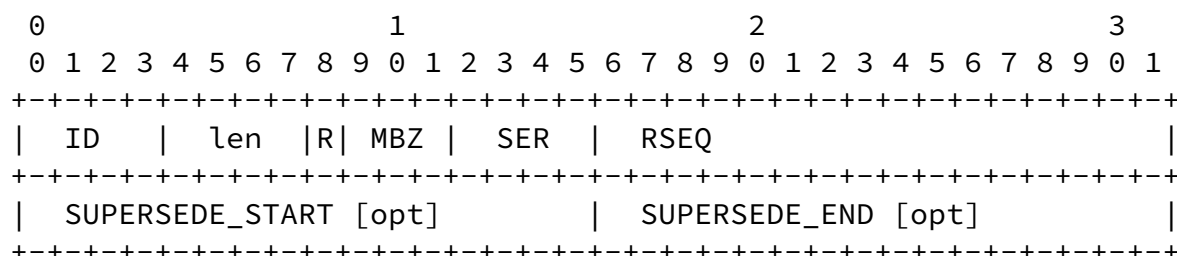


Figure 1: R packet header extension element

The structure of this header extension element is illustrated in Figure 1. Its fields are defined as follows:

- ID: 4 bits The local identifier negotiated for this header extension element, as defined in [[I-D.ietf-avt-rtp-hdext](#)].
- Length (len): 4 bits The length minus one of the data bytes of this header extension element, not counting the header byte (ID and len), as defined in [[I-D.ietf-avt-rtp-hdext](#)]. This will have the value 6 if the second word (the superseded range) is present, and 2 if it is not. Its value MUST either be 2 or 6.
- R: 1 bit A bit indicating that the packet containing this header extension element is an R packet. If this bit is set (1), this header extension element identifies the current packet as an R packet in series SER with R sequence number RSEQ. If it is not set (0), this header extension element instead indicates that the media stream's most recent R packet in series SER had R sequence number RSEQ. (This latter case is known as a "mark element".) An R packet is a packet where exactly one header extension element has the R bit set, whereas a non-R packet is a packet where no header header extension element has the R bit set. If this bit is not set, the superseded range SHOULD NOT be present for that header extension element (i.e. the len field SHOULD be 2) and MUST be ignored if present.
- Reserved, Must Be Zero (MBZ): 3 bits Reserved bits. These MUST be set to zero on transmit and ignored on receive.
- Series ID (SER): 4 bits An identifier of which series of R packets is being described by this header extension element. If a media encoder is describing only a single series of R packets, this SHOULD have the value 0.

R Packet Sequence Number (RSEQ): 16 bits An unsigned sequence number indicating the number of this R packet within the series SER. This value is incremented by 1 (modulo 2^{16}) for every R packet sent in a given series. RSEQ values for separate series are independent.

Start of Superseded Range (SUPERSEDE_START): 16 bits The R sequence number of the earliest R packet, inclusive, superseded by this R packet, calculated modulo 2^{16} . (Since this value uses modulo

arithmetic, the value $RSEQ + 1$ MAY be used for `SUPERSEDE_START` to indicate that all R packets prior to the end of the superseded range have been superseded.) This field is optional, and is only present when `len=6`.

End of Superseded Range (`SUPERSEDE_END`): 16 bits The R sequence number of the final R packet, inclusive, superseded by this R packet, calculated modulo 2^{16} . This value MUST lie in the closed range [`SUPERSEDE_START` .. `RSEQ`] modulo 2^{16} . This field is optional, and is only present when `len=6`.

An RTP packet MAY contain multiple R packet mark elements, so long as each of these elements has a different value for `SER`. However, an RTP MUST NOT contain more than one of these header extension elements with the R bit set, i.e. an R packet may not belong to more than one series.

All RTP packets in a media stream using R packets SHOULD include a mark element for all active series.

When the second word of this header extension element is present, it indicates that this R packet supersedes some previously-received R packets, meaning that these packets are no longer necessary in order to reconstruct stream state. This second word MUST only appear in a header extension element which has its R bit set.

An R packet can only supersede R packets in the series identified by the element's `SER` field. R packets cannot supersede packets in other series.

It is valid for a supersede element to have `SUPERSEDE_END=RSEQ`. This indicates that the R packet supersedes itself, i.e. that this R packet immediately becomes irrelevant to the stream state. The most common reason to do this would be to end a series; this can be done by sending an empty packet (e.g. an RTP No-op [[I-D.ietf-avt-rtp-no-op](https://tools.ietf.org/html/draft-ietf-avt-rtp-no-op)] packet) with the superseded range (`SUPERSEDE_START`, `SUPERSEDE_END`) = ($RSEQ+1$, `RSEQ`), so that the series no longer contains any non-superseded packets.

The first R packet sent in a series SHOULD be sent with a superseded range such that `SUPERSEDE_START` = $RSEQ+1$, to make it clear that no

previous R packets are present in the range.

R packets MAY redundantly include already-superseded packets in their range of packets to be superseded. If a group of superseding R packets are sent together (e.g., if a number of packets together encode an intra frame), they SHOULD all be marked as superseding the same range of R packets.

6. RTCP Feedback Messages

This section describes the RTCP feedback message used to indicate that R packets have been lost.

The R Packet Negative Acknowledgement (RNACK) Message is an RTCP Feedback (AVPF) [[I-D.ietf-avt-rtcp-feedback](#)] message identified by PT=RTPFB and FMT=4 (value tentatively chosen). The FCI field MUST contain at least one and MAY contain more than one RNACK.

The RNACK packet is used to indicate the loss of one or more R packets. The lost packet(s) are identified by means of a packet sequence number, the series identifier, and a bit mask.

The structure and semantics of the RNACK message are similar to that of the AVPF Generic NACK message.

The RNACK Feedback Control Information (FCI) field has the following syntax Figure 2:

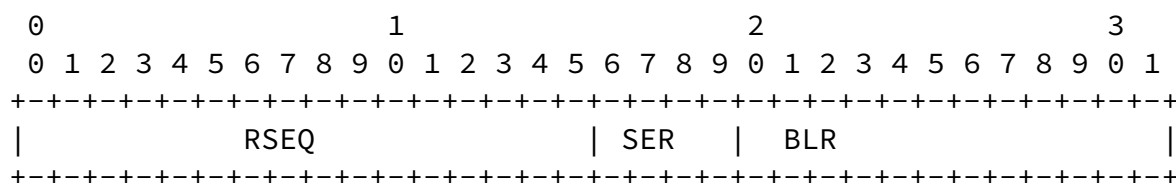


Figure 2: Feedback Control Information for R Packet Negative Acknowledgment (RNACK)

R Packet Sequence Number (RSEQ): 16 bits The RSEQ field indicates a RSEQ value that the receiver has not received.

Series ID (SER): 4 bits An identifier of which series of R packets is being described as being lost by this header extension element.

Bitmask of following Lost R Packets (BLR): 12 bits The BLR allows for reporting losses of any of the 12 R Packets packets immediately following the RTP packet indicated by RSEQ. Denoting the BLR's least significant bit as bit 1, and its most significant bit as bit 12, then bit i of the bit mask is set to 1 if the receiver has not received R packet number $(RSEQ+i)$ in the series SER (modulo

2^16) and indicates this packet is lost; bit *i* is set to 0 otherwise. Note that the sender MUST NOT assume that a receiver has received an R packet because its bit mask was set to 0. For example, the least significant bit of the BLR would be set to 1 if the packet corresponding to RSEQ and the following R packet in the series had been lost. However, the sender cannot infer that packets RSEQ+2 through RSEQ+16 have been received simply because bits 2 through 15 of the BLR are 0; all the sender knows is that the receiver has not reported them as lost at this time.

When a receiver detects that it has not received a non-superseded R packet, it sends an RNACK message as soon as possible, subject to the rules of RTCP feedback [[I-D.ietf-avt-rtcp-feedback](#)]. (In multipoint scenarios, this includes listening for RNACK packets from other receivers and not sending an RNACK for a lost R packet that has already been reported.)

When a sender receives an RNACK packet, it checks whether the packet has been superseded. If it has not, it retransmits the packet for which an RNACK was sent (using, e.g., the RTP retransmission payload [[I-D.ietf-avt-rtp-retransmission](#)]). If the packet has been superseded, it retransmits the most recent packet whose R packet element indicated a superseded packet range including the packet requested.

A sender MAY choose to generate and send a new R packet superseding the one requested in an RNACK, rather than retransmitting a packet that has been sent previously.

If, after some period of time, a receiver has not received either a retransmission of the R packet for which an RNACK was sent, or an R packet superseding that packet, it SHOULD retransmit the RNACK message. A receiver MUST NOT send RNACK messages more often than permitted by AVPF. It SHOULD perform estimation of the round-trip time to the sender, if possible, and SHOULD NOT send RNACK messages more often than once per round-trip time. (If the receiver is also acting as an RTP sender, and the sender is sending RTCP reception reports for the receiver's stream, round-trip times can be inferred from the sender report's LSR and DLSR fields.) If the round-trip time is not available, receivers SHOULD NOT send RNACK messages more often than once per 100 milliseconds. [TODO: this value is a complete guess.]

[7.](#) Security Considerations

All security considerations of RTP [[RFC3550](#)] apply here as well.

A receiver MUST NOT assume that an R packet bitstream it receives actually provides complete decoder state; the bitstream MUST still be validated.

[8.](#) IANA Considerations

This document defines an org.ietf RTP header extension element: 'org.ietf.r-packet/200606'. Its format is defined in [Section 5](#) . This header extension element should be registered by IANA as described in [[I-D.ietf-avt-rtp-hdext](#)].

This document defines an RTCP transport-layer feedback message: 'RNACK'. Its format is defined in [Section 6](#). It should be defined in the RTPFB FMT as follows:

Name: RNACK

Long name: R Packet Negative Acknowledgment

Value: 4 (tentatively chosen)

Reference: RFC XXXX.

[9.](#) References

[9.1.](#) Normative References

[[I-D.ietf-avt-rtcp-feedback](#)]

Ott, J. and S. Wenger, "Extended RTP Profile for RTCP-based Feedback(RTP/AVPF)", [draft-ietf-avt-rtcp-feedback-11](#) (work in progress), August 2004.

[[I-D.ietf-avt-rtp-hdext](#)]

Singer, D., "A general mechanism for RTP Header Extensions", [draft-ietf-avt-rtp-hdext-03](#) (work in progress), June 2006.

[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.

[9.2.](#) Informative References

[I-D.ietf-avt-rtp-no-op]
Andreasen, F., "A No-Op Payload Format for RTP",
[draft-ietf-avt-rtp-no-op-00](#) (work in progress), May 2005.

Lennox

Expires December 21, 2006

[Page 10]

Internet-Draft

Recoverable High-Priority RTP Packets

June 2006

[I-D.ietf-avt-rtp-retransmission]
Rey, J., "RTP Retransmission Payload Format",
[draft-ietf-avt-rtp-retransmission-12](#) (work in progress),
September 2005.

[I-D.ietf-avt-ulp]
Li, A., "RTP Payload Format for Generic Forward Error
Correction", [draft-ietf-avt-ulp-17](#) (work in progress),
March 2006.

[ITU.H264.2005]
International Telecommunications Union, "Advanced video
coding for generic audiovisual services", ITU-
T Recommendation H.264 (2005), ISO Standard 14496-10:2005,
March 2005.

[RFC2833] Schulzrinne, H. and S. Petrack, "RTP Payload for DTMF
Digits, Telephony Tones and Telephony Signals", [RFC 2833](#),
May 2000.

[RFC3984] Wenger, S., Hannuksela, M., Stockhammer, T., Westerlund,
M., and D. Singer, "RTP Payload Format for H.264 Video",
[RFC 3984](#), February 2005.

[Appendix A.](#) Requirements

This appendix describes the requirements motivating the design of the protocol described in the rest of this document. (This appendix is not normative; it thus does not use the formal upper-case versions of the [RFC 2119](#) [[RFC2119](#)] terms for its requirements.)

An encoder must be able to indicate a subset of the packets in a generated RTP stream as being high-priority (R) packets.

A decoder must be able to determine when it has lost R packets, whenever any packet of the stream is received, and regardless of the dependency structure of the encoded stream.

Similarly, a decoder must be able to determine that it has not lost any R packets as of the latest packet that has been received, regardless of how many other non-R packets have been lost.

An encoder must be able to split a frame into any number of R packets, either in a codec-aware manner (e.g. H.264 [[ITU.H264.2005](#)] slices) or a codec-unaware manner (e.g. [RFC 3984](#) [[RFC3984](#)] fragmentation units).

An encoder must be able to state that an R packet supersedes previous R packets, i.e. that some previous R packets are no longer necessary in order to establish the stream state. This includes both being able to state that all R packets before a given one have been superseded, and that a range of R packets are superseded.

It must be possible for an encoder to apply forward error correction [[I-D.ietf-avt-ulp](#)] to its media stream, either to all packets or selectively only to R packets, in a way that allows R packet state to be recovered from the FEC stream.

An encoder must be able to describe multiple separate series of R packets, such that an R packet may supersede all previous R packets of one series without altering the need for other series. It must also be able to describe the most recent R packet of multiple series.

[Appendix B](#). Analysis of Alternate Approaches

This appendix describes some alternate approaches that could be considered to provide a reliable substream of a media stream. It describes how they fail to satisfy the requirements listed in [Appendix A](#).

[B.1](#). Generic NACK

This approach uses the Generic NACK mechanism [I-D.ietf-avt-rtcp-feedback]. A receiver of a stream sends a Generic NACK for every RTP packet it misses. The stream sender then re-sends only those packets it knows are needed for the reliable subset of the stream.

In addition to causing many more NACK messages to be sent than are necessary for the stream, this approach's primary difficulty is that there is no way for a receiver to know when it should stop sending NACK messages for those packets which it will not receive.

[B.2.](#) Generic NACK with Codec knowledge

This approach is like the previous one, except that decoders track codec-specific dependency state to infer whether a missed packet was an R packet or not, and they only send Generic NACK messages for the packets that they conclude were R packets.

This conclusion is not in general possible, if a receiver misses both an R packet and a subsequent non-R packet directly dependent on it. Additionally, if an R frame is fragmented or sliced, a decoder can't determine which packets were part of the sliced or fragmented R frame.

[B.3.](#) Codec-Specific NACK

This approach is also like the previous one, except that decoders send codec-specific NACK messages (in RPSI messages) listing specific parts of frames that have been missed.

It is not in general possible to tell whether a missed frame was an R packet, and the layering mechanisms don't support media-independent fragmentation (for example [RFC 3984](#) [RFC3984] fragmentation units).

[B.4.](#) Multiple streams

In this approach, R packets are sent on a separate RTP session than the the non-R packets. RTP NACK messages are sent for packets on the R packet session only.

This approach has the drawback that receipt of a non-R packet does not trigger a NACK request for a missed R packet; only the next R

packet does that, unless the receiver has knowledge of the sender's timing of R packets and thus can pro-actively send a NACK for a packet it does not actually know it has missed. Since R packets tend to be spaced far apart, this can greatly delay stream recovery.

Author's Address

Jonathan Lennox
Layered Media, Inc.
350 W. Passaic St
Fourth Floor
Rochelle Park, NJ 07662
US

Email: [jonathan at layeredmedia.com](mailto:jonathan@layeredmedia.com)

Intellectual Property Statement

The IETF takes no position regarding the validity or scope of any Intellectual Property Rights or other rights that might be claimed to

pertain to the implementation or use of the technology described in this document or the extent to which any license under such rights might or might not be available; nor does it represent that it has made any independent effort to identify any such rights. Information on the procedures with respect to rights in RFC documents can be found in [BCP 78](#) and [BCP 79](#).

Copies of IPR disclosures made to the IETF Secretariat and any assurances of licenses to be made available, or the result of an attempt made to obtain a general license or permission for the use of such proprietary rights by implementers or users of this specification can be obtained from the IETF on-line IPR repository at <http://www.ietf.org/ipr>.

The IETF invites any interested party to bring to its attention any copyrights, patents or patent applications, or other proprietary rights that may cover technology that may be required to implement this standard. Please address the information to the IETF at ietf-ipr@ietf.org.

Disclaimer of Validity

This document and the information contained herein are provided on an "AS IS" basis and THE CONTRIBUTOR, THE ORGANIZATION HE/SHE REPRESENTS OR IS SPONSORED BY (IF ANY), THE INTERNET SOCIETY AND THE INTERNET ENGINEERING TASK FORCE DISCLAIM ALL WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY WARRANTY THAT THE USE OF THE INFORMATION HEREIN WILL NOT INFRINGE ANY RIGHTS OR ANY IMPLIED WARRANTIES OF MERCHANTABILITY OR FITNESS FOR A PARTICULAR PURPOSE.

Copyright Statement

Copyright (C) The Internet Society (2006). This document is subject to the rights, licenses and restrictions contained in [BCP 78](#), and except as set forth therein, the authors retain all their rights.

Acknowledgment

Funding for the RFC Editor function is currently provided by the Internet Society.