Internet Engineering Task Force
INTERNET-DRAFT
draft-kohler-dccp-mobility-00.ps
Expires: January 2005
Eddie Kohler
UCLA
12 July 2004

Datagram Congestion Control Protocol Mobility and Multihoming

Status of this Memo

This document is an Internet-Draft.

By submitting this Internet-Draft, we certify that any applicable patent or other IPR claims of which we are aware have been disclosed, or will be disclosed, and any of which we become aware will be disclosed, in accordance with RFC 3668 (BCP 79).

By submitting this Internet-Draft, we accept the provisions of Section 3 of RFC 3667 (BCP 78).

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 a "work in progress."

The list of current Internet-Drafts can be accessed at http://www.ietf.org/1id-abstracts.html

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

Copyright Notice

Copyright (C) The Internet Society (2004). All Rights Reserved.

Abstract

This document lays out requirements and a preliminary design for transport-level mobility and multihoming support in the Datagram Congestion Control Protocol (DCCP) [DCCP].

Table of Contents

1. Introduction

The Datagram Congestion Control Protocol (DCCP) [DCCP] did not originally contain support for mobility or multihoming. Each DCCP connection was associated with exactly two network-level addresses over its lifetime, one per endpoint. At the very first DCCP BoF session, at the 51st IETF in London, we got some feedback criticizing this decision, so we added mobility and multihoming support as of November 2001. This decision was supported by the eventual DCCP working group charter:

> Prior to the final development of the protocol, the working group will investigate areas of functionality that should be integrated into DCCP because they are difficult or impossible to layer above it. These areas include security and multi-homing/mobility, at a minimum.

Thus began our quest for a mechanism that would support mobility and multihoming, at the DCCP level, with reasonable security and DoS-prevention, without using cryptography. (The DCCP working group's charter has been interpreted as preventing DCCP from including cryptography, even MD5 hashes.) DCCP's mobility support changed, often fundamentally, in every succeeding draft.

Unsurprisingly, we did not find a suitable mechanism, and I now believe no such mechanism exists. Even seemingly trivial multihoming mechanisms like SCTP's can cause security problems in practice [ANC04]. Mobility and multihoming have removed from the main DCCP specification.

Unfortunately, mobility and multihoming support can't easily be implemented at a higher-level layer, and there are good arguments for supporting mobility and multihoming at the transport layer -- not least required interactions with congestion control. This document, therefore, presents one potential design for DCCP mobility and multihoming support. It relaxes one of DCCP mobility's original requirements by using cryptography.

2. Requirements

DCCP mobility and multihoming support should fulfill the following requirements and non-requirements.

- An endpoint does not need to announce a new address before moving to that address.

  RATIONALE: Mobile hosts usually cannot know a new address before a move completes. Multihomed hosts *can* know each of their addresses, but announcing addresses before using them does not prevent all attacks; see, for example, the "address squatting" attacks in [ANC04].

- Move requests must be safe against hijacking. Even attackers that can snoop on part or all of data traffic must not be able to move a connection. However, move requests need not be safe against man-in-the-middle attackers with control over which packets get delivered (such as routers).

  RATIONALE: Moving a connection is in some ways the worst possible attack: An attacker takes over a user's identity, without the user becoming aware of the attack or being able to stop the attack. We must prevent this. However, an endpoint with full control over the path could carry out this kind of attack even without mobility support. An attacker that can snoop a packet sent by the mobile host, then prevent that snooped packet from being delivered to the stationary host, can subvert our mobility mechanism.

- Mobility must not create new, large opportunities for denial-of-service attacks.

- Endpoints must be able to move freely between different NAT domains using the mobility mechanism.
- Simultaneous moves need not be supported.
- Cryptography is allowed.

It is difficult, perhaps impossible, to fulfill both the NAT traversal and hijacking prevention requirements. Natural mechanisms for preventing hijacking, such as cryptographically signing the packet's network headers, fail in the presence of NATs, which change those headers. NATs essentially hijack connections by definition; we want to allow that, but prevent malicious hijacking. The solution below represents one attempt.

3. Design Overview

The mobility design described here has the following features.

0. Support for mobility is optional and defaults to off.
1. Each endpoint of a mobility-capable connection has a public 128-bit Mobility ID.
2. The endpoints share a Mobility Secret, a key communicated over a secure channel. The Secret is either transmitted out-of-band, or via public-key cryptography or Diffie-Hellman exchange. It is changed on every successful move.
3. A Mobility Sequence number increases monotonically with moves, and identifies which Mobility Secret a packet is using.

The two mobility identifiers, Mobility ID and Mobility Secret, are used for different purposes. Mobility ID lets the stationary endpoint (and any NATs on the stationary endpoint's side) map a move announcement to an existing connection, even though the source address and port have changed. (The original source address and port cannot be used for this purpose, since NATs in the original path may cause the two endpoints to disagree on their values.) Mobility Secret is used during the move handshake to prevent most hijackings.

Moving a connection progresses through the following four steps, which will take approximately two round-trip times.

1. The mobile host moves, then sends a DCCP-Move-Request packet from its new address. This packet contains (1) the stationary host's Mobility ID, in the clear, and (2) a mobility token, encrypted by the Mobility Secret. Each mobility token consists of a portion of the DCCP header (Sequence Number, Acknowledgement Number, Mobility ID, and so forth) and a new half-Mobility Secret.

   This packet serves to inform the stationary endpoint of the move.

2. The stationary host maps the DCCP-Move-Request packet to the old connection using the Mobility ID, then checks the token against the packet data. If this check succeeds, it sends a DCCP-Move-Response packet to the DCCP-Move-Request packet's source address, including a similar token. The stationary host remembers both the new Mobility Secret and the old Mobility Secret.

   Together, the initial DCCP-Move-Request and this Response define a new Mobility Secret, which is used as a nonce. This ensures that (1) Mobility Secrets are not terribly vulnerable to attack, since they frequently change, and (2) the mobile endpoint receives partial confirmation that its packets are getting through to the stationary endpoint.

3. The mobile host sends a DCCP-Move-Confirm packet. This time, the token is encrypted with the new Mobility Secret. The token's New Mobility Sequence and

half-secret fields equal those from the DCCP-MoveRequest. Receiving the DCCP-Move-Response packet informs the mobile host that the stationary host received its move request.

This packet proves to the stationary endpoint that the true mobile endpoint received its DCCP-Move-Response. This is because the token is encrypted with the new Secret, which could only have been obtained by a party that new the old Secret.

4. The stationary host receives the DCCP-Move-Confirm packet, removes its old Mobility Secret(s), and sends a DCCP-Move-Complete packet back to the sender, both to end this episode of mobility and to inform NATs and middleboxes that the connection's endpoints have definitively changed. The token's New Mobility Sequence and half-secret fields equal those from the DCCP-MoveResponse.

4. Protocol Details

DCCP mobility and multihoming support requires one more packet type, DCCP-Move; two new DCCP features, Mobility Capable (server-priority boolean, defaults to 0) and Mobility ID (non-negotiable 128-bit number); one new Reset Code, "Move Refused"; and two new connection properties, Mobility Sequence and Mobility Secret. Additionally, any plausible mechanism for establishing a Mobility Secret in-band would also use features to do it.

4.1. DCCP-Move Header

Mobility handshakes take place using the DCCP-Move packet type. DCCP A sends a DCCP-Move packet to DCCP B after changing its address and/or port number. The DCCP-Move packet requests that DCCP B start sending packets to a new address and port number, which are read off the packet's network header and generic DCCP header. The old address and port are defined through the Mobility ID.

```
 0                   1                   2                   3
 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 8 9 0 1
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
/              Generic DCCP Header (12 or 16 bytes)             /
/                    with Type=8 (DCCP-Move)                    /
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|   Reserved    |            Acknowledgement Number             |
(+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+)X=1
(.       Acknowledgement Number (low bits)       |   Reserved    |)only
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|    Subtype    |   Reserved    |       Mobility Sequence       |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                    Mobility ID (high bits)                    .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                   Mobility ID (bits 64-95)                    .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                   Mobility ID (bits 32-63)                    .
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
.                    Mobility ID (low bits)                     |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                     Options                     /    Padding    |
+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+=+
|                       Encrypted Message                       |
|                              ...                              |
```

Subtype: 8 bits
Used to distinguish between different message types in the mobility handshake.

Mobility Sequence: 16 bits
Defines the Mobility Secret used to encrypt the token. Its initial value for the connection is 0.

Mobility ID: 128 bits
The value of the receiver's Mobility ID feature. This value uniquely identifies the current connection among the set of connections terminating at the receiver (meaning, the stationary endpoint); it MUST have been set in an earlier exchange. See Section 5.2.

The application data area of a DCCP-Move packet contains a Mobility Token, a message encrypted by the Mobility Secret. This message consists of the following fields, packed together:

```
  1. The third, fourth, and subsequent words of the DCCP
     header, up to, but not including, any options.
     This includes the Sequence Number, Acknowledgement
     Number, Type, Subtype, Mobility Sequence, and
     Mobility ID fields, among others.
  2. New Mobility Sequence.
  3. New half-Mobility Secret (format not specified yet).
4-6. Same as 1-3.
```

An endpoint MUST ignore any DCCP-Move packet whose application data area doesn't have the proper format.

Regarding sequence-validity, DCCP-Move Sequence and Acknowledgement Numbers are not strongly checked because moves might likely happen after long loss periods, and the mandatory Mobility ID provides good protection against unexpected packets. A DCCP-Move packet is sequence-valid if "seqno >= SWL" and "GAR <= ackno <= AWH". Sequence-invalid DCCP-Move packets MUST be ignored.

5. Multihoming and Mobility

THIS SECTION IS NOT IN ITS FINAL FORM. SOME TEXT IS OLD.

DCCP provides primitive support for multihoming and mobility via a mechanism for transferring a connection endpoint from one address to another. The moving endpoint must negotiate mobility support beforehand. When the moving endpoint gets a new address, it sends a DCCP-Move packet from that address to the stationary endpoint. The stationary endpoint then changes its connection state to use the new address.

DCCP's support for mobility is intended to solve only the simplest multihoming and mobility problems; for instance, there's no support for simultaneous moves. Applications requiring more complex mobility semantics, or more stringent security guarantees, should use an existing solution like Mobile IP or [SB00]. DCCP mobility may not be useful in the context of IPv6, with its mandatory support for Mobile IP.

5.1. Mobility Capable Feature

A DCCP uses the Mobility Capable feature to inform its partner that it would like to be able to change its address and/or port during the course of the connection. DCCP B sends a "Change R(Mobility Capable, 1)" option to DCCP A to inform it that B might like to move later.

Mobility Capable has feature number , and is server-priority. It takes one-byte Boolean values. DCCP A agrees in principle to accept DCCP-Move packets from DCCP B when Mobility Capable/A is one. DCCP A MUST reject any DCCP-Move packet for a connection whose Mobility Capable/A feature is zero, although it MAY reject a valid DCCP-Move packet even when Mobility Capable/A is one. Values of two or more are reserved. New connections start with Mobility Capable 0 (that is, mobility is not allowed) for both endpoints.

5.2. Mobility ID Feature

A DCCP uses the Mobility ID feature to inform its partner of a 128-bit number that will act as identification, should the partner change its address and/or port during the course of the connection. DCCP A sends a "Change L(Mobility ID, N)" option to notify DCCP B of the ID it has chosen for B's use.

Mobility ID has feature number , and is non-negotiable. Its values are sixteen-byte integers. The Mobility ID/A feature equals the identifier that DCCP B should use on DCCP-Move packets sent to A. DCCP A chooses Mobility ID/A to uniquely identify the connection among all connections that terminate at A. For security, DCCP A MUST choose Mobility ID/A randomly. New connections start with Mobility ID 0 for both endpoints. However, Mobility IDs of zero MUST NOT be accepted on DCCP-Move packets; an endpoint cannot successfully move until the relevant Mobility ID has been set to a nonzero value.

5.3. Mobile Host Processing

When DCCP A changes its address and/or port, it MUST signal this by sending DCCP B a DCCP-Move-Request packet with enclosed Mobility Token. Of special interest is the New Mobility Sequence in the token. The Mobility Sequence for each new connection is 0. Thereafter, it increases monotonically as endpoints try to move. Every time DCCP A sends a DCCP-Move-Request from a new address, it MUST increment New Mobility Sequence.

Once DCCP A receives a DCCP-Move-Response, it calculates the new Mobility Secret from the fragments stored in the two tokens; throws away all prior Mobility Secrets; sets its Mobility Sequence and Secret fields from the packet; and sends a DCCP-Move-Confirm, confirming that it has done these things.

DCCP A SHOULD retransmit its DCCP-Move-Request and -Confirm packets until it receives some confirmation. The retransmission strategy SHOULD be similar to that for retransmitting DCCP-Requests; for instance, a first timeout on the order of a second, with an exponential backoff timer.

DCCP A MUST reset its congestion control state after sending a DCCP-Move, since nothing is known about conditions on the new path. Essentially, DCCP A must "slow start" up to its new fair rate, as appropriate for its congestion control mechanism.

DCCP A SHOULD NOT send non-DCCP-Move packets to DCCP B until the move is confirmed. If it did so, and the DCCP-Move packet was lost or reordered, then DCCP B would react by sending DCCP-Resets with Reset Code 3, "No Connection". DCCP A might implement special handling for such resets to avoid any post-move quiet period, but this is NOT RECOMMENDED.

DCCP B MAY refuse to accept a move, perhaps because of address policy. In this case, DCCP A will receive a DCCP-Reset with Reset Code "Move Refused", rather than a confirming DCCP-Sync. DCCP A MAY react by tearing down the connection, or by trying another DCCP-Move -- for instance, back to the old address, if possible.

DCCP endpoints SHOULD NOT use an old address-port pair after sending a DCCP-Move. If it becomes necessary to switch back to the old address-port pair, the endpoint MUST do so explicitly using another DCCP-Move.

DCCP-Move packets SHOULD NOT be sent until the connection is established; it is illegal to send a DCCP-Move in REQUEST or RESPOND state. If an endpoint moves during connection establishment, it SHOULD abandon the old connection and initiate a new one. No connection exists to move until the three-way handshake has completed.

5.4. Stationary Host Processing

The stationary endpoint, DCCP B, uses DCCP-Move packets' destination address, destination port, and Mobility ID fields to look up the relevant connection. This differs from all other packet types, which use the source address/source port/destination address/destination port 4-tuple.

DCCP B MUST ignore DCCP-Moves whose Mobility ID is zero, or whose Mobility ID does not correspond to any active connection. It also MUST ignore DCCP-Moves sent to sockets in CLOSED, LISTEN, REQUEST, RESPOND, or TIMEWAIT state, and it MUST ignore DCCP-Moves with invalid Sequence or Acknowledgement Numbers, Mobility Sequences, Subtypes, or Mobility Tokens. DCCP B MUST NOT respond to invalid DCCP-Moves with DCCP-Reset or DCCP-Sync packets, since any active response would leak information about the connection to a possibly malicious host. After receiving an invalid DCCP-Move, DCCP B MAY ignore subsequent DCCP-Move packets, valid or not, for a short period of time, such as one second or one round-trip time. This protects DCCP B against denial-of-service attacks from floods of invalid DCCP-Moves.

On receiving a valid DCCP-Move-Request or DCCP-Move-Confirm, DCCP B first checks the Mobility Sequence and Secret against the network address. If two packets are ever received that have (1) the same Mobility Sequence; (2) valid tokens, indicating possession of a valid Mobility Secret for that Sequence; and (3) different network addresses, then this is a strong indication of potential attack. The connection MUST be reset, and all state purged.

After this check, DCCP B will respond to a valid DCCP-Move-Request by (1) constructing a new half-key for Mobility Secret; (2) storing the new Mobility Secret under the suggested Mobility Sequence (but it also keeps the old Secret); (3) sending a DCCP-Move-Response, encrypted using the old Secret.

DCCP B will respond to a valid DCCP-Move-Confirm by (1) deleting all Mobility Secrets whose Sequences are lower than the confirmed Sequence, and (2) sending a DCCP-Move-Complete, encrypted using the new Secret.

5.5. Congestion Control State

Once an endpoint has transitioned to a new address, the connection is effectively a new connection in terms of its congestion control state: the accumulated information about congestion between the old endpoints no longer applies. Both DCCPs MUST initialize their congestion control state (windows, rates, and so forth) to that of a new connection. That is, they must "slow start".

Similarly, the endpoints' PMTUs SHOULD be reinitialized, and PMTU discovery performed again, following an address change.

During the transition period between addresses, the endpoints might receive congestion feedback from both before the move and after the move. Congestion and loss events on packets sent before the move SHOULD NOT affect the new connection's congestion control state.

5.6. Security Considerations

TBA

Normative References

[DCCP] E. Kohler, M. Handley, and S. Floyd. Datagram Congestion Control Protocol, draft-ietf-dccp-spec-07.txt, work in progress, July 2004.

[RFC 793] J. Postel, editor. Transmission Control Protocol. RFC 793.

[RFC 1191] J. C. Mogul and S. E. Deering. Path MTU Discovery. RFC 1191.

[RFC 1750] D. Eastlake, S. Crocker, and J. Schiller. Randomness Recommendations for Security. RFC 1750.

[RFC 2119] S. Bradner. Key Words For Use in RFCs to Indicate Requirement Levels. RFC 2119.

[RFC 2460] S. Deering and R. Hinden. Internet Protocol, Version 6 (IPv6) Specification. RFC 2460.

[RFC 3168] K.K. Ramakrishnan, S. Floyd, and D. Black. The Addition of Explicit Congestion Notification (ECN) to IP. RFC 3168.

[RFC 3309] J. Stone, R. Stewart, and D. Otis. Stream Control Transmission Protocol (SCTP) Checksum Change. RFC 3309.

[RFC 3692] T. Narten. Assigning Experimental and Testing Numbers Considered Useful. RFC 3692.

[UDP-LITE] L-A. Larzon, M. Degermark, S. Pink, L-E. Jonsson (editor), and G. Fairhurst (editor). The UDP-Lite Protocol. draft-ietf-tsvwg-udp-lite-02.txt, work in progress, August 2003.

Informative References

[ANC04] Tuomas Aura, Pekka Nikander, and Gonzalo Camarillo. Effects of Mobility and Multihoming on Transport-Protocol Security. 2004 IEEE Symposium Security and Privacy.

[BB01] S.M. Bellovin and M. Blaze. Cryptographic Modes of Operation for the Internet. 2nd NIST Workshop on Modes of Operation, August 2001.

[BEL98] S.M. Bellovin. Cryptography and the Internet. Proc. CRYPTO '98 (LNCS 1462), pp46-55, August, 1988.

[CCID 2 PROFILE] S. Floyd and E. Kohler. Profile for DCCP Congestion Control ID 2: TCP-like Congestion Control. draft-ietf-dccp-ccid2-05.txt, work in progress, February 2004.

[CCID 3 PROFILE] S. Floyd, E. Kohler, and J. Padhye. Profile for DCCP Congestion Control ID 3: TFRC Congestion Control. draft-ietf-dccp-ccid3-05.txt, work in progress, February 2004.

[LINK BCP] Phil Karn, editor. Advice for Internet Subnetwork Designers. draft-ietf-pilc-link-design-13.txt, work in progress, February 2003.

[M85] Robert T. Morris. A Weakness in the 4.2BSD Unix TCP/IP Software. Computer Science Technical Report 117, AT&T Bell Laboratories, Murray Hill, NJ, February 1985.

[PMTUD] Matt Mathis, John Heffner, and Kevin Lahey. Path MTU Discovery. draft-ietf-pmtud-method-00.txt, work in progress, October 2003.

[RFC 792] J. Postel, editor. Internet Control Message Protocol. RFC 792.

[RFC 1948] S. Bellovin. Defending Against Sequence Number Attacks. RFC 1948.

[RFC 2960] R. Stewart, Q. Xie, K. Morneault, C. Sharp, H. Schwarzbauer, T. Taylor, I. Rytina, M. Kalla, L. Zhang, and V. Paxson. Stream Control Transmission Protocol. RFC 2960.

[RFC 3124] H. Balakrishnan and S. Seshan. The Congestion Manager. RFC 3124.

[RFC 3360] S. Floyd. Inappropriate TCP Resets Considered Harmful. RFC 3360.

[RFC 3448] M. Handley, S. Floyd, J. Padhye, and J. Widmer. TCP Friendly Rate Control (TFRC): Protocol Specification. RFC 3448.

[RFC 3517] E. Blanton, M. Allman, K. Fall, and L. Wang. A Conservative Selective Acknowledgment (SACK)-based Loss Recovery Algorithm for TCP. RFC 3517.

[RFC 3540] N. Spring, D. Wetherall, and D. Ely. Robust Explicit Congestion Notification (ECN) Signaling with Nonces. RFC 3540.

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

[SB00] Alex C. Snoeren and Hari Balakrishnan. An End-to-End Approach to Host Mobility. Proc. 6th Annual ACM/IEEE International Conference on Mobile Computing and Networking (MOBICOM '00), August 2000.

[SHHP00] Oliver Spatscheck, Jorgen S. Hansen, John H. Hartman, and Larry L. Peterson. Optimizing TCP Forwarder Performance. IEEE/ACM Transactions on Networking 8(2):146-157, April 2000.

[SYNCOOKIES] Daniel J. Bernstein. SYN Cookies. http://cr.yp.to/syncookies.html, as of July 2003.

Authors' Addresses

Eddie Kohler <kohler@cs.ucla.edu>
4531C Boelter Hall
UCLA Computer Science Department
Los Angeles, CA 90095
USA

Full Copyright Statement

Copyright (C) The Internet Society 2004. 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.

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.

Intellectual Property

The IETF has been notified of intellectual property rights claimed in regard to some or all of the specification contained in this document. For more information consult the online list of claimed rights.

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.