                   Negotiation of NAT-Traversal in the IKE

Status of this Memo

   This document specifies an Internet standards track protocol for the
   Internet community, and requests discussion and suggestions for
   improvements. Please refer to the current edition of the "Internet
   Official Protocol Standards" (STD 1) for the standardization state
   and status of this protocol. Distribution of this memo is unlimited.

Copyright Notice

Abstract

   This document describes how to detect one or more network address
   translation devices (NATs) between IPsec hosts, and how to negotiate
   the use of UDP encapsulation of IPsec packets through NAT boxes in
   Internet Key Exchange (IKE).

Table of Contents

1. Introduction

    This document is split into two parts. The first describes what is
    needed in IKE Phase 1 for NAT-Traversal support. This includes
    detecting whether the other end supports NAT-Traversal, and detecting
    whether there is one or more NATs between the peers.

    The second part describes how to negotiate the use of UDP
    encapsulated IPsec packets in IKE's Quick Mode. It also describes
    how to transmit the original source and destination addresses to the
    peer, if required. These addresses are used in transport mode to
    update the TCP/IP checksums incrementally so that they will match
    after the NAT transform. (The NAT cannot do this, because the TCP/IP
    checksum is inside the UDP encapsulated IPsec packet.)

    The document [RFC3948] describes the details of UDP encapsulation,
    and [RFC3715] provides background information and motivation of NAT-
    Traversal in general. In combination with [RFC3948], this document
    represents an "unconditionally compliant" solution to the
    requirements as defined by [RFC3715].

    In the basic scenario for this document, the initiator is behind
    NA(P)T, and the responder has a fixed static IP address.

This document defines a protocol that will work even if both ends are
behind NAT, but the process of how to locate the other end is out of
the scope of this document. In one scenario, the responder is behind
a static host NAT (only one responder per IP, as there is no way to
use any destination ports other than 500/4500). That is, it is known
by the configuration.

2. Specification of Requirements

This document shall use the keywords "MUST", "MUST NOT", "REQUIRED",
"SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED, "MAY",
and "OPTIONAL" to describe requirements. They are to be interpreted
as described in [RFC2119].

3. Phase 1

The detection of support for NAT-Traversal and detection of NAT along
the path between the two IKE peers occurs in IKE [RFC2409] Phase 1.

The NAT may change the IKE UDP source port, and recipients MUST be
able to process IKE packets whose source port is different from 500.
The NAT does not have to change the source port if:

o only one IPsec host is behind the NAT, or

o for the first IPsec host, the NAT can keep the port 500, and the
   NAT will only change the port number for later connections.

Recipients MUST reply back to the source address from the packet (see
[RFC3715], section 2.1, case d). This means that when the original
responder is doing rekeying or sending notifications to the original
initiator, it MUST send the packets using the same set of port and IP
numbers used when the IKE SA was last used.

For example, when the initiator sends a packet with source and
destination port 500, the NAT may change it to a packet with source
port 12312 and destination port 500. The responder must be able to
process the packet whose source port is 12312. It must reply back
with a packet whose source port is 500 and destination port is 12312.
The NAT will then translate this packet to source port 500 and
destination port 500.

3.1. Detecting Support of NAT-Traversal

   The NAT-Traversal capability of the remote host is determined by an
   exchange of vendor ID payloads. In the first two messages of Phase
   1, the vendor id payload for this specification MUST be sent if
   supported (and it MUST be received by both sides) for the NAT-
   Traversal probe to continue. The content of the payload is the MD5
   hash of

      RFC 3947

   The exact content in hex for the payload is

      4a131c81070358455c5728f20e95452f

3.2. Detecting the Presence of NAT

   The NAT-D payload not only detects the presence of NAT between the
   two IKE peers, but also detects where the NAT is. The location of
   the NAT device is important, as the keepalives have to initiate from
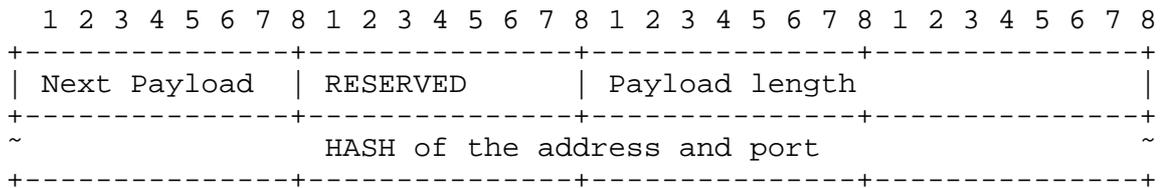   the peer "behind" the NAT.

   To detect NAT between the two hosts, we have to detect whether the IP
   address or the port changes along the path. This is done by sending
   the hashes of the IP addresses and ports of both IKE peers from each
   end to the other. If both ends calculate those hashes and get same
   result, they know there is no NAT between. If the hashes do not
   match, somebody has translated the address or port. This means that
   we have to do NAT-Traversal to get IPsec packets through.

   If the sender of the packet does not know his own IP address (in case
   of multiple interfaces, and the implementation does not know which IP
   address is used to route the packet out), the sender can include
   multiple local hashes to the packet (as separate NAT-D payloads). In
   this case, NAT is detected if and only if none of the hashes match.

   The hashes are sent as a series of NAT-D (NAT discovery) payloads.
   Each payload contains one hash, so in case of multiple hashes,
   multiple NAT-D payloads are sent. In the normal case there are only
   two NAT-D payloads.

   The NAT-D payloads are included in the third and fourth packets of
   Main Mode, and in the second and third packets in the Aggressive
   Mode.

The format of the NAT-D packet is

```
    1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
   +---------------+---------------+---------------+---------------+
   | Next Payload  | RESERVED      | Payload length                |
   +---------------+---------------+---------------+---------------+
   ~                   HASH of the address and port               ~
   +---------------+---------------+---------------+---------------+
```

The payload type for the NAT discovery payload is 20.

The HASH is calculated as follows:

        HASH = HASH(CKY-I | CKY-R | IP | Port)

This uses the negotiated HASH algorithm. All data inside the HASH is
in the network byte-order. The IP is 4 octets for an IPv4 address
and 16 octets for an IPv6 address. The port number is encoded as a 2
octet number in network byte-order. The first NAT-D payload contains
the remote end's IP address and port (i.e., the destination address
of the UDP packet). The remaining NAT-D payloads contain possible
local-end IP addresses and ports (i.e., all possible source addresses
of the UDP packet).

If there is no NAT between the peers, the first NAT-D payload
received should match one of the local NAT-D payloads (i.e., the
local NAT-D payloads this host is sending out), and one of the other
NAT-D payloads must match the remote end's IP address and port. If
the first check fails (i.e., first NAT-D payload does not match any
of the local IP addresses and ports), it means that there is dynamic
NAT between the peers, and this end should start sending keepalives
as defined in the [RFC3948] (this end is behind the NAT).

The CKY-I and CKY-R are the initiator and responder cookies. They
are added to the hash to make precomputation attacks for the IP
address and port impossible.

The following example is of a Phase 1 exchange using NAT-Traversal in
Main Mode (authentication with signatures):

```
Initiator                              Responder
-----------                            -----------
HDR, SA, VID -->
                                       <-- HDR, SA, VID
HDR, KE, Ni, NAT-D, NAT-D -->
                                       <-- HDR, KE, Nr, NAT-D, NAT-D
HDR*#, IDii, [CERT, ] SIG_I -->
                                       <-- HDR*#, IDir, [CERT, ], SIG_R
```

The following example is of Phase 1 exchange using NAT-Traversal in
Aggressive Mode (authentication with signatures):

```
Initiator                              Responder
-----------                            -----------
HDR, SA, KE, Ni, IDii, VID -->
                                       <-- HDR, SA, KE, Nr, IDir,
                                               [CERT, ], VID, NAT-D,
                                               NAT-D, SIG_R
HDR*#, [CERT, ], NAT-D, NAT-D,
    SIG_I -->
```

The # sign indicates that those packets are sent to the changed port
if NAT is detected.

4. Changing to New Ports

   IPsec-aware NATs can cause problems (See [RFC3715], section 2.3).
   Some NATs will not change IKE source port 500 even if there are
   multiple clients behind the NAT (See [RFC3715], section 2.3, case n).
   They can also use IKE cookies to demultiplex traffic instead of using
   the source port (See [RFC3715], section 2.3, case m). Both of these
   are problematic for generic NAT transparency, as it is difficult for
   IKE to discover the capabilities of the NAT. The best approach is
   simply to move the IKE traffic off port 500 as soon as possible to
   avoid any IPsec-aware NAT special casing.

   Take the common case of the initiator behind the NAT. The initiator
   must quickly change to port 4500 once the NAT has been detected to
   minimize the window of IPsec-aware NAT problems.

   In Main Mode, the initiator MUST change ports when sending the ID
   payload if there is NAT between the hosts. The initiator MUST set
   both UDP source and destination ports to 4500. All subsequent
   packets sent to this peer (including informational notifications)
   MUST be sent on port 4500. In addition, the IKE data MUST be
   prepended with a non-ESP marker allowing for demultiplexing of
   traffic, as defined in [RFC3948].

   Thus, the IKE packet now looks like this:

        IP UDP(4500,4500) <non-ESP marker> HDR*, IDii, [CERT,] SIG_I

   This assumes authentication using signatures. The 4 bytes of non-ESP
   marker are defined in the [RFC3948].

When the responder gets this packet, the usual decryption and
processing of the various payloads is performed. If these are
successful, the responder MUST update local state so that all
subsequent packets (including informational notifications) to the
peer use the new port, and possibly the new IP address obtained from
the incoming valid packet. The port will generally be different, as
the NAT will map UDP(500,500) to UDP(X,500), and UDP(4500,4500) to
UDP(Y,4500). The IP address will seldom be different from the pre-
changed IP address. The responder MUST respond with all subsequent
IKE packets to this peer by using UDP(4500,Y).

Similarly, if the responder has to rekey the Phase 1 SA, then the
rekey negotiation MUST be started by using UDP(4500,Y). Any
implementation that supports NAT traversal MUST support negotiations
that begin on port 4500. If a negotiation starts on port 4500, then
it doesn't need to change anywhere else in the exchange.

Once port change has occurred, if a packet is received on port 500,
that packet is old. If the packet is an informational packet, it MAY
be processed if local policy allows this. If the packet is a Main
Mode or an Aggressive Mode packet (with the same cookies as previous
packets), it SHOULD be discarded. If the packet is a new Main Mode
or Aggressive exchange, then it is processed normally (the other end
might have rebooted, and this is starting new exchange).

Here is an example of a Phase 1 exchange using NAT-Traversal in Main
Mode (authentication with signatures) with changing port:

```
Initiator                         Responder
-----------                       -----------
UDP(500,500) HDR, SA, VID -->
                                  <-- UDP(500,X) HDR, SA, VID

UDP(500,500) HDR, KE, Ni,
    NAT-D, NAT-D -->
                                  <-- UDP(500,X) HDR, KE, Nr,
                                          NAT-D, NAT-D

UDP(4500,4500) HDR*#, IDii,
    [CERT, ]SIG_I -->
                                  <-- UDP(4500,Y) HDR*#, IDir,
                                          [ CERT, ], SIG_R
```

The procedure for Aggressive Mode is very similar. After the NAT has
been detected, the initiator sends IP UDP(4500,4500) <4 bytes of
non-ESP marker> HDR*, [CERT,], NAT-D, NAT-D, and SIG_I. The
responder does similar processing to the above, and if successful,
MUST update it's internal IKE ports. The responder MUST respond with
all subsequent IKE packets to this peer by using UDP(4500,Y).

```
    Initiator                           Responder
    -----------                         ------------
UDP(500,500) HDR, SA, KE,
    Ni, IDii, VID -->

                                    <-- UDP(500,X) HDR, SA, KE,
                                            Nr, IDir, [CERT, ],
                                            VID, NAT-D, NAT-D,
                                            SIG_R
UDP(4500,4500) HDR*#, [CERT, ],
    NAT-D, NAT-D,
    SIG_I -->

                                    <-- UDP(4500, Y) HDR*#, ...
```

If the support of the NAT-Traversal is enabled, the port in the ID
payload in Main Mode/Aggressive Mode MUST be set to 0.

The most common case for the responder behind the NAT is if the NAT
is simply doing 1:1 address translation. In this case, the initiator
still changes both ports to 4500. The responder uses an algorithm
identical to that above, although in this case Y will equal 4500, as
no port translation is happening.

A different port change case involves out-of-band discovery of the
ports to use. Those discovery methods are out of the scope of this
document. For instance, if the responder is behind a port
translating NAT, and the initiator needs to contact it first, then
the initiator will have to determine which ports to use, usually by
contacting some other server. Once the initiator knows which ports
to use to traverse the NAT, generally something like UDP(Z,4500), it
initiates using these ports. This is similar to the responder rekey
case above in that the ports to use are already known up front, and
no additional change has to take place. Also, the first keepalive
timer starts after the change to the new port, and no keepalives are
sent to the port 500.

5. Quick Mode

   After Phase 1, both ends know whether there is a NAT present between
   them. The final decision of using NAT-Traversal is left to Quick
   Mode. The use of NAT-Traversal is negotiated inside the SA payloads
   of Quick Mode. In Quick Mode, both ends can also send the original
   addresses of the IPsec packets (in case of the transport mode) to the
   other end so that each can fix the TCP/IP checksum field after the
   NAT transformation.

5.1. Negotiation of the NAT-Traversal Encapsulation

   The negotiation of the NAT-Traversal happens by adding two new
   encapsulation modes. These encapsulation modes are

   UDP-Encapsulated-Tunnel   3
   UDP-Encapsulated-Transport   4

   It is not normally useful to propose both normal tunnel or transport
   mode and UDP-Encapsulated modes. UDP encapsulation is required to
   fix the inability to handle non-UDP/TCP traffic by NATs (see
   [RFC3715], section 2.2, case i).

   If there is a NAT box between hosts, normal tunnel or transport
   encapsulations may not work. In this case, UDP-Encapsulation SHOULD
   be used.

   If there is no NAT box between, there is no point in wasting
   bandwidth by adding UDP encapsulation of packets. Thus, UDP-
   Encapsulation SHOULD NOT be used.

   Also, the initiator SHOULD NOT include both normal tunnel or
   transport mode and UDP-Encapsulated-Tunnel or UDP-Encapsulated-
   Transport in its proposals.

5.2. Sending the Original Source and Destination Addresses

   To perform incremental TCP checksum updates, both peers may need to
   know the original IP addresses used by their peers when those peers
   constructed the packet (see [RFC3715], section 2.1, case b). For the
   initiator, the original Initiator address is defined to be the
   Initiator's IP address. The original Responder address is defined to
   be the perceived peer's IP address. For the responder, the original
   Initiator address is defined to be the perceived peer's address. The
   original Responder address is defined to be the Responder's IP
   address.

   The original addresses are sent by using NAT-OA (NAT Original
   Address) payloads.

   The Initiator NAT-OA payload is first. The Responder NAT-OA payload
   is second.

   Example 1:

          Initiator <---------> NAT <---------> Responder
               ^   ^   ^
             Iaddr   NatPub   Raddr

The initiator is behind a NAT talking to the publicly available
responder. Initiator and Responder have the IP addresses Iaddr and
Raddr. NAT has public IP address NatPub.

Initiator:

                    NAT-OAi = Iaddr
                    NAT-OAr = Raddr

Responder:

                    NAT-OAi = NATPub
                    NAT-OAr = Raddr

Example 2:

        Initiator <------> NAT1 <---------> NAT2 <-------> Responder
               ^   ^   ^   ^
            Iaddr   Nat1Pub   Nat2Pub   Raddr

Here, NAT2 "publishes" Nat2Pub for Responder and forwards all traffic
to that address to Responder.

Initiator:

                    NAT-OAi = Iaddr
                    NAT-OAr = Nat2Pub

Responder:

                    NAT-OAi = Nat1Pub
                    NAT-OAr = Raddr

In the case of transport mode, both ends MUST send both original
Initiator and Responder addresses to the other end. For tunnel mode,
both ends SHOULD NOT send original addresses to the other end.

The NAT-OA payloads are sent inside the first and second packets of
Quick Mode. The initiator MUST send the payloads if it proposes any
UDP-Encapsulated-Transport mode, and the responder MUST send the
payload only if it selected UDP-Encapsulated-Transport mode. It is
possible that the initiator sends the NAT-OA payload but proposes
both UDP-Encapsulated transport and tunnel mode. Then the responder
selects the UDP-Encapsulated tunnel mode and does not send the NAT-OA
payload back.

The format of the NAT-OA packet is

```
        1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
       +---------------+---------------+---------------+---------------+
       | Next Payload  | RESERVED      | Payload length                |
       +---------------+---------------+---------------+---------------+
       | ID Type       | RESERVED      | RESERVED                      |
       +---------------+---------------+---------------+---------------+
       |              IPv4 (4 octets) or IPv6 address (16 octets)      |
       +---------------+---------------+---------------+---------------+
```

The payload type for the NAT original address payload is 21.

The ID type is defined in the [RFC2407]. Only ID_IPV4_ADDR and
ID_IPV6_ADDR types are allowed. The two reserved fields after the ID
Type must be zero.

The following example is of Quick Mode using NAT-OA payloads:

```
Initiator                              Responder
------------                           ------------
HDR*, HASH(1), SA, Ni, [, KE]
    [, IDci, IDcr ]
    [, NAT-OAi, NAT-OAr] -->
                             <-- HDR*, HASH(2), SA, Nr, [, KE]
                                        [, IDci, IDcr ]
                                        [, NAT-OAi, NAT-OAr]

HDR*, HASH(3) -->
```

6. Initial Contact Notifications

   The source IP and port address of the INITIAL-CONTACT notification
   for the host behind NAT are not meaningful (as NAT can change them),
   so the IP and port numbers MUST NOT be used to determine which
   IKE/IPsec SAs to remove (see [RFC3715], section 2.1, case c). The ID
   payload sent from the other end SHOULD be used instead; i.e., when an
   INITIAL-CONTACT notification is received from the other end, the
   receiving end SHOULD remove all the SAs associated with the same ID
   payload.

7. Recovering from the Expiring NAT Mappings

   There are cases where NAT box decides to remove mappings that are
   still alive (for example, when the keepalive interval is too long, or
   when the NAT box is rebooted). To recover from this, ends that are
   NOT behind NAT SHOULD use the last valid UDP encapsulated IKE or
   IPsec packet from the other end to determine which IP and port
   addresses should be used. The host behind dynamic NAT MUST NOT do

this, as otherwise it opens a DoS attack possibility because the IP
address or port of the other host will not change (it is not behind
NAT).

Keepalives cannot be used for these purposes, as they are not
authenticated, but any IKE authenticated IKE packet or ESP packet can
be used to detect whether the IP address or the port has changed.

8. Security Considerations

Whenever changes to some fundamental parts of a security protocol are
proposed, the examination of security implications cannot be skipped.
Therefore, here are some observations about the effects, and about
whether or not these effects matter.

   o IKE probes reveal NAT-Traversal support to anyone watching the
      traffic. Disclosing that NAT-Traversal is supported does not
      introduce new vulnerabilities.

   o The value of authentication mechanisms based on IP addresses
      disappears once NATs are in the picture. That is not necessarily
      a bad thing (for any real security, authentication measures other
      than IP addresses should be used). This means that authentication
      with pre-shared keys cannot be used in Main Mode without using
      group-shared keys for everybody behind the NAT box. Using group
      shared keys is a huge risk because it allows anyone in the group
      to authenticate to any other party and claim to be anybody in the
      group; e.g., a normal user could impersonate a vpn-gateway and act
      as a man in the middle, and read/modify all traffic to/from others
      in the group. Use of group-shared keys is NOT RECOMMENDED.

   o As the internal address space is only 32 bits and is usually very
      sparse, it might be possible for the attacker to find out the
      internal address used behind the NAT box by trying all possible
      IP-addresses to find the matching hash. The port numbers are
      normally fixed to 500, and the cookies can be extracted from the
      packet. This limits the hash calculations to 2^{32}. If an
      educated guess of the private address space is made, then the
      number of hash calculations needed to find out the internal IP
      address goes down to $2^{24} + 2 * (2^{16})$.

   o Neither NAT-D payloads nor Vendor ID payloads are authenticated in
      Main Mode nor in Aggressive Mode. This means that attacker can
      remove those payloads, modify them, or add them. By removing or
      adding them, the attacker can cause Denial of Service attacks. By
      modifying the NAT-D packets, the attacker can cause both ends to
      use UDP-Encapsulated modes instead of directly using tunnel or
      transport mode, thus wasting some bandwidth.

o Sending the original source address in the Quick Mode reveals the
   internal IP address behind the NAT to the other end. In this case
   we have already authenticated the other end, and sending the
   original source address is only needed in transport mode.

o Updating the IKE SA/ESP UDP encapsulation IP addresses and ports
   for each valid authenticated packet can cause DoS if an attacker
   can listen to all traffic in the network, change the order of the
   packets, and inject new packets before the packet he has already
   seen. In other words, the attacker can take an authenticated
   packet from the host behind NAT, change the packet UDP source or
   destination ports or IP addresses and send it out to the other end
   before the real packet reaches it. The host not behind the NAT
   will update its IP address and port mapping and send further
   traffic to the wrong host or port. This situation is fixed
   immediately when the attacker stops modifying the packets, as the
   first real packet will fix the situation. Implementations SHOULD
   AUDIT the event every time the mapping is changed, as it should
   not happen that often.

9. IANA Considerations

   This document contains two new "magic numbers" allocated from the
   existing IANA registry for IPsec and renames existing registered port
   4500. This document also defines 2 new payload types for IKE.

   The following are new items that have been added in the "Internet
   Security Association and Key Management Protocol (ISAKMP)
   Identifiers" Encapsulation Mode registry:

Name	Value	Reference
UDP-Encapsulated-Tunnel	3	[RFC3947]
UDP-Encapsulated-Transport	4	[RFC3947]

   Change in the registered port registry:

Keyword	Decimal	Description	Reference
ipsec-nat-t	4500/tcp	IPsec NAT-Traversal	[RFC3947]
ipsec-nat-t	4500/udp	IPsec NAT-Traversal	[RFC3947]

New IKE payload numbers need to be added to the Next Payload Types
registry:

```
NAT-D          20              NAT Discovery Payload
NAT-OA         21              NAT Original Address Payload
```

10. IAB Considerations

The UNSAF [RFC3424] questions are addressed by the IPsec-NAT
compatibility requirements document [RFC3715].

11. Acknowledgments

Thanks to Markus Stenberg, Larry DiBurro, and William Dixon, who
contributed actively to this document.

Thanks to Tatu Ylonen, Santeri Paavolainen, and Joern Sierwald, who
contributed to the document used as the base for this document.

12. References

12.1. Normative References

   [RFC2409] Harkins, D. and D. Carrel, "The Internet Key Exchange
             (IKE)", RFC 2409, November 1998.

   [RFC2407] Piper, D., "The Internet IP Security Domain of
             Interpretation for ISAKMP", RFC 2407, November 1998.

   [RFC3948] Huttunen, A., Swander, B., Volpe, V., DiBurro, L., and M.
             Stenberg, "UDP Encapsulation of IPsec Packets", RFC 3948,
             January 2005.

   [RFC2119] Bradner, S., "Key words for use in RFCs to Indicate
             Requirement Levels", BCP 14, RFC 2119, March 1997.

12.2. Informative References

   [RFC3715] Aboba, B. and W. Dixon, "IPsec-Network Address Translation
             (NAT) Compatibility Requirements", RFC 3715, March 2004.

   [RFC3424] Daigle, L. and IAB, "IAB Considerations for UNilateral
             Self-Address Fixing (UNSAF) Across Network Address
             Translation", RFC 3424, November 2002.

Authors' Addresses

    Tero Kivinen
    SafeNet, Inc.
    Fredrikinkatu 47
    FIN-00100 HELSINKI
    Finland

    EMail: kivinen@safenet-inc.com

    Ari Huttunen
    F-Secure Corporation
    Tammasaarenkatu 7,
    FIN-00181 HELSINKI
    Finland

    EMail: Ari.Huttunen@F-Secure.com

    Brian Swander
    Microsoft
    One Microsoft Way
    Redmond, WA 98052
    USA

    EMail: briansw@microsoft.com

    Victor Volpe
    Cisco Systems
    124 Grove Street
    Suite 205
    Franklin, MA 02038
    USA

    EMail: vvolpe@cisco.com

Full Copyright Statement

   Copyright (C) The Internet Society (2005).

   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 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 IETF's procedures with respect to rights in IETF 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.

Acknowledgement

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