                    UDP Encapsulation of IPsec ESP Packets

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.

Abstract

   This protocol specification defines methods to encapsulate and
   decapsulate IP Encapsulating Security Payload (ESP) packets inside
   UDP packets for traversing Network Address Translators.  ESP
   encapsulation, as defined in this document, can be used in both IPv4
   and IPv6 scenarios.  Whenever negotiated, encapsulation is used with
   Internet Key Exchange (IKE).

Table of Contents

1.  Introduction

   This protocol specification defines methods to encapsulate and
   decapsulate ESP packets inside UDP packets for traversing Network
   Address Translators (NATs) (see [RFC3715], section 2.2, case i).  The
   UDP port numbers are the same as those used by IKE traffic, as
   defined in [RFC3947].

   The sharing of the port numbers for both IKE and UDP encapsulated ESP
   traffic was selected because it offers better scaling (only one NAT
   mapping in the NAT; no need to send separate IKE keepalives), easier
   configuration (only one port to be configured in firewalls), and
   easier implementation.

   A client's needs should determine whether transport mode or tunnel
   mode is to be supported (see [RFC3715], Section 3, "Telecommuter
   scenario").  L2TP/IPsec clients MUST support the modes as defined in
   [RFC3193].  IPsec tunnel mode clients MUST support tunnel mode.

   An IKE implementation supporting this protocol specification MUST NOT
   use the ESP SPI field zero for ESP packets.  This ensures that IKE
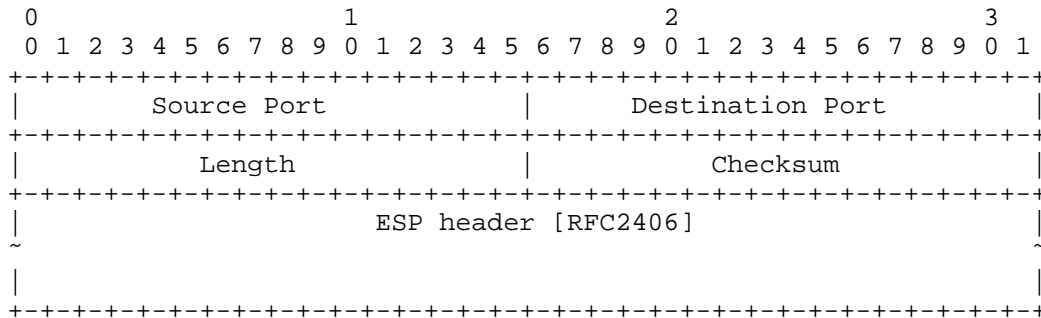   packets and ESP packets can be distinguished from each other.

   As defined in this document, UDP encapsulation of ESP packets is
   written in terms of IPv4 headers.  There is no technical reason why
   an IPv6 header could not be used as the outer header and/or as the
   inner header.

   Because the protection of the outer IP addresses in IPsec AH is
   inherently incompatible with NAT, the IPsec AH was left out of the
   scope of this protocol specification.  This protocol also assumes
   that IKE (IKEv1 [RFC2401] or IKEv2 [IKEv2]) is used to negotiate the
   IPsec SAs.  Manual keying is not supported.

   The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT",
   "SHOULD", "SHOULD NOT", "RECOMMENDED",  "MAY", and "OPTIONAL" in this
   document are to be interpreted as described in [RFC2119].

2.  Packet Formats

2.1.  UDP-Encapsulated ESP Header Format

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|          Source Port          |       Destination Port        |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|            Length             |           Checksum            |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
|                      ESP header [RFC2406]                     |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
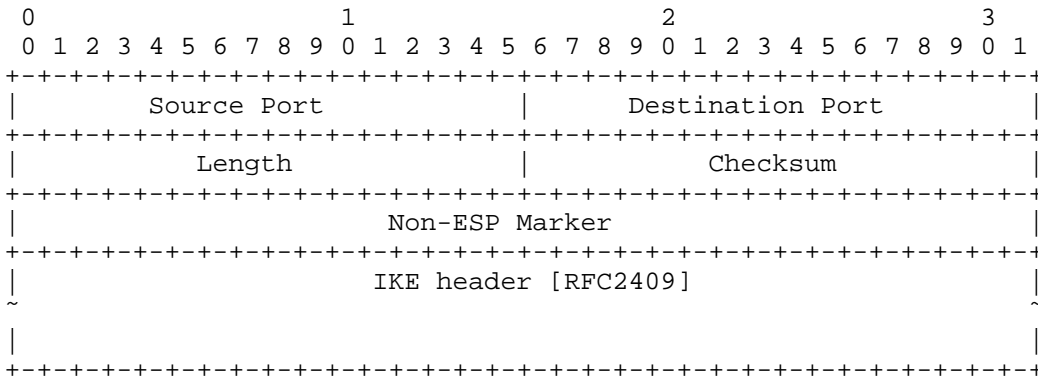
   The UDP header is a standard [RFC0768] header, where

   o   the Source Port and Destination Port MUST be the same as that used
       by IKE traffic,
   o   the IPv4 UDP Checksum SHOULD be transmitted as a zero value, and
   o   receivers MUST NOT depend on the UDP checksum being a zero value.

   The SPI field in the ESP header MUST NOT be a zero value.
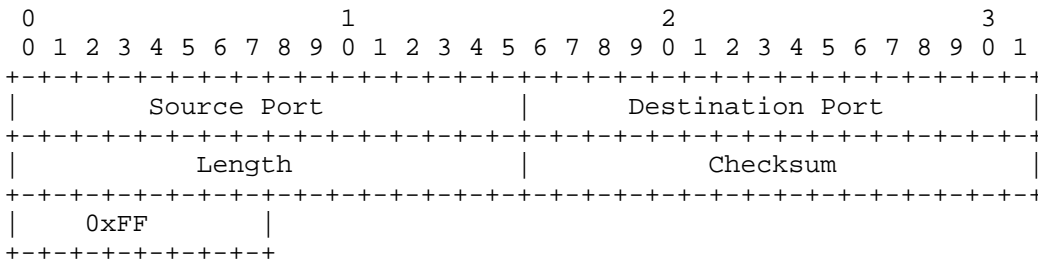
2.2.  IKE Header Format for Port 4500

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Source Port          |       Destination Port        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Length              |           Checksum            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                        Non-ESP Marker                         |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |                       IKE header [RFC2409]                    |
   ~                                                               ~
   |                                                               |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

   The UDP header is a standard [RFC0768] header and is used as defined
   in [RFC3947].  This document does not set any new requirements for
   the checksum handling of an IKE packet.

   A Non-ESP Marker is 4 zero-valued bytes aligning with the SPI field
   of an ESP packet.

2.3.  NAT-Keepalive Packet Format

```
    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
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |          Source Port          |       Destination Port        |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |           Length              |           Checksum            |
   +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   |    0xFF       |
   +-+-+-+-+-+-+-+-+
```

   The UDP header is a standard [RFC0768] header, where

   o   the Source Port and Destination Port MUST be the same as used by
       UDP-ESP encapsulation of Section 2.1,
   o   the IPv4 UDP Checksum SHOULD be transmitted as a zero value, and
   o   receivers MUST NOT depend upon the UDP checksum being a zero
       value.

   The sender MUST use a one-octet-long payload with the value 0xFF.
   The receiver SHOULD ignore a received NAT-keepalive packet.

3.  Encapsulation and Decapsulation Procedures

3.1.  Auxiliary Procedures

3.1.1.  Tunnel Mode Decapsulation NAT Procedure

   When a tunnel mode has been used to transmit packets (see [RFC3715],
   section 3, criteria "Mode support" and "Telecommuter scenario"), the
   inner IP header can contain addresses that are not suitable for the
   current network.  This procedure defines how these addresses are to
   be converted to suitable addresses for the current network.

   Depending on local policy, one of the following MUST be done:

   1.  If a valid source IP address space has been defined in the policy
       for the encapsulated packets from the peer, check that the source
       IP address of the inner packet is valid according to the policy.
   2.  If an address has been assigned for the remote peer, check that
       the source IP address used in the inner packet is the assigned IP
       address.
   3.  NAT is performed for the packet, making it suitable for transport
       in the local network.

3.1.2.  Transport Mode Decapsulation NAT Procedure

   When a transport mode has been used to transmit packets, contained
   TCP or UDP headers will have incorrect checksums due to the change of
   parts of the IP header during transit.  This procedure defines how to
   fix these checksums (see [RFC3715], section 2.1, case b).

   Depending on local policy, one of the following MUST be done:
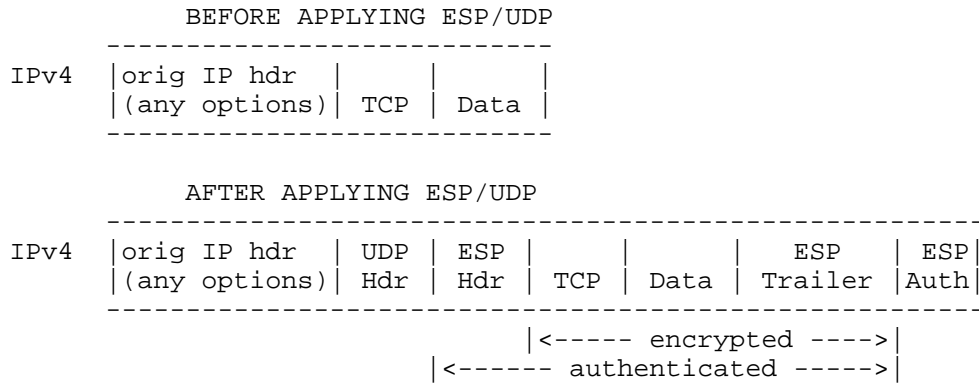
   1.  If the protocol header after the ESP header is a TCP/UDP header
       and the peer's real source and destination IP address have been
       received according to [RFC3947], incrementally recompute the
       TCP/UDP checksum:

       *  Subtract the IP source address in the received packet from the
          checksum.
       *  Add the real IP source address received via IKE to the
          checksum (obtained from the NAT-OA)
       *  Subtract the IP destination address in the received packet
          from the checksum.
       *  Add the real IP destination address received via IKE to the
          checksum (obtained from the NAT-OA).
       Note: If the received and real address are the same for a given
       address (e.g., say the source address), the operations cancel and
       don't need to be performed.

2.  If the protocol header after the ESP header is a TCP/UDP header,
    recompute the checksum field in the TCP/UDP header.

3.  If the protocol header after the ESP header is a UDP header, set
    the checksum field to zero in the UDP header.  If the protocol
    after the ESP header is a TCP header, and if there is an option
    to flag to the stack that the TCP checksum does not need to be
    computed, then that flag MAY be used.  This SHOULD only be done
    for transport mode, and if the packet is integrity protected.
    Tunnel mode TCP checksums MUST be verified.  (This is not a
    violation to the spirit of section 4.2.2.7 in [RFC1122] because a
    checksum is being generated by the sender and verified by the
    receiver.  That checksum is the integrity over the packet
    performed by IPsec.)

In addition an implementation MAY fix any contained protocols that
have been broken by NAT (see [RFC3715], section 2.1, case g).

3.2.  Transport Mode ESP Encapsulation

```
            BEFORE APPLYING ESP/UDP
            --------------------------
  IPv4  |orig IP hdr  |     |      |
        |(any options)| TCP | Data |
        --------------------------

            AFTER APPLYING ESP/UDP
        -------------------------------------------------------
  IPv4  |orig IP hdr  | UDP | ESP |     |      |    ESP   | ESP|
        |(any options)| Hdr | Hdr | TCP | Data | Trailer |Auth|
        -------------------------------------------------------
                                  |<----- encrypted ---->|
                            |<------ authenticated ----->|
```
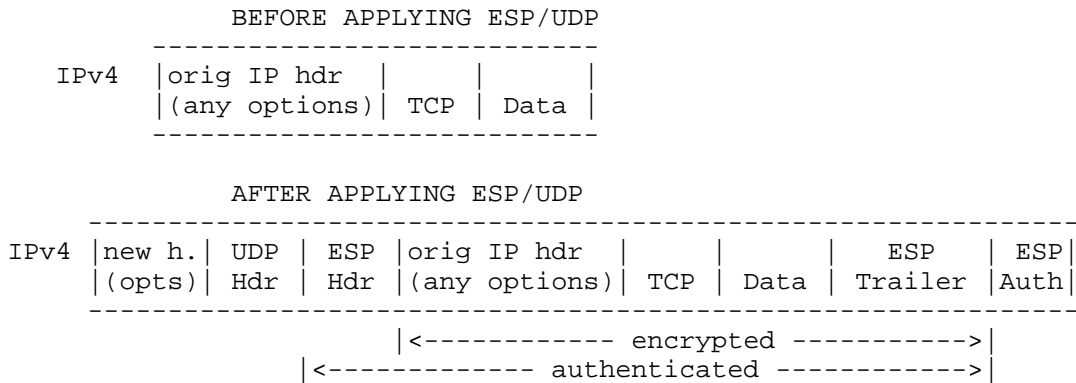
1.  Ordinary ESP encapsulation procedure is used.
2.  A properly formatted UDP header is inserted where shown.
3.  The Total Length, Protocol, and Header Checksum (for IPv4) fields
    in the IP header are edited to match the resulting IP packet.

3.3.  Transport Mode ESP Decapsulation

1.  The UDP header is removed from the packet.
2.  The Total Length, Protocol, and Header Checksum (for IPv4) fields
    in the new IP header are edited to match the resulting IP packet.
3.  Ordinary ESP decapsulation procedure is used.
4.  Transport mode decapsulation NAT procedure is used.

3.4.  Tunnel Mode ESP Encapsulation

```
                 BEFORE APPLYING ESP/UDP
              ----------------------------
      IPv4    |orig IP hdr  |     |       |
              |(any options)| TCP | Data  |
              ----------------------------

                 AFTER APPLYING ESP/UDP
       ------------------------------------------------------------------
 IPv4 |new h.| UDP  | ESP |orig IP hdr  |     |       |   ESP   | ESP|
      |(opts)| Hdr  | Hdr |(any options)| TCP | Data  | Trailer |Auth|
       ------------------------------------------------------------------
                                  |<----------- encrypted ----------->|
                            |<------------ authenticated ------------>|
```

   1.  Ordinary ESP encapsulation procedure is used.
   2.  A properly formatted UDP header is inserted where shown.
   3.  The Total Length, Protocol, and Header Checksum (for IPv4) fields
   in the new IP header are edited to match the resulting IP packet.

3.5.  Tunnel Mode ESP Decapsulation

   1.  The UDP header is removed from the packet.
   2.  The Total Length, Protocol, and Header Checksum (for IPv4) fields
       in the new IP header are edited to match the resulting IP packet.
   3.  Ordinary ESP decapsulation procedure is used.
   4.  Tunnel mode decapsulation NAT procedure is used.

4.  NAT Keepalive Procedure

   The sole purpose of sending NAT-keepalive packets is to keep NAT
   mappings alive for the duration of a connection between the peers
   (see [RFC3715], Section 2.2, case j).  Reception of NAT-keepalive
   packets MUST NOT be used to detect whether a connection is live.
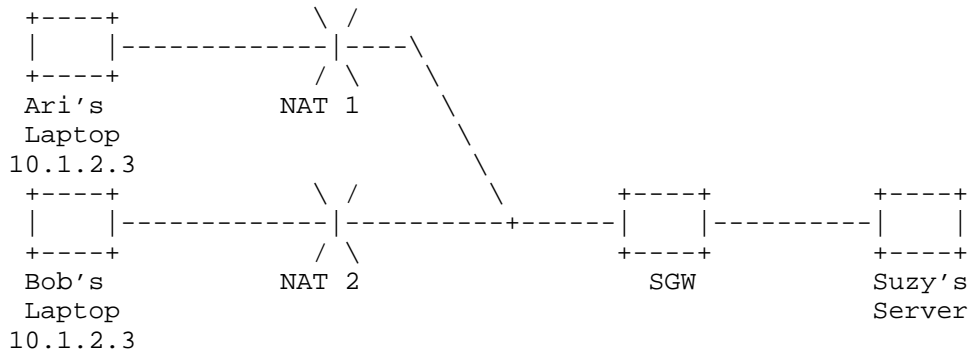
   A peer MAY send a NAT-keepalive packet if one or more phase I or
   phase II SAs exist between the peers, or if such an SA has existed at
   most N minutes earlier.  N is a locally configurable parameter with a
   default value of 5 minutes.

   A peer SHOULD send a NAT-keepalive packet if a need for it is
   detected according to [RFC3947] and if no other packet to the peer
   has been sent in M seconds.  M is a locally configurable parameter
   with a default value of 20 seconds.

5.  Security Considerations

5.1.  Tunnel Mode Conflict

   Implementors are warned that it is possible for remote peers to
   negotiate entries that overlap in an SGW (security gateway), an issue
   affecting tunnel mode (see [RFC3715], section 2.1, case e).

```
           +----+              \ /
           |    |------------|----\
           +----+             / \    \
           Ari's           NAT 1      \
           Laptop                      \
           10.1.2.3                     \
           +----+              \ /       \      +----+          +----+
           |    |------------|----------+------|    |----------|    |
           +----+             / \                +----+          +----+
           Bob's           NAT 2                 SGW            Suzy's
           Laptop                                               Server
           10.1.2.3
```

   Because SGW will now see two possible SAs that lead to 10.1.2.3, it
   can become confused about where to send packets coming from Suzy's
   server.  Implementors MUST devise ways of preventing this from
   occurring.

   It is RECOMMENDED that SGW either assign locally unique IP addresses
   to Ari's and Bob's laptop (by using a protocol such as DHCP over
   IPsec) or use NAT to change Ari's and Bob's laptop source IP
   addresses to these locally unique addresses before sending packets
   forward to Suzy's server.  This covers the "Scaling" criteria of
   section 3 in [RFC3715].

   Please see Appendix A.

5.2.  Transport Mode Conflict

   Another similar issue may occur in transport mode, with 2 clients,
   Ari and Bob, behind the same NAT talking securely to the same server
   (see [RFC3715], Section 2.1, case e).

   Cliff wants to talk in the clear to the same server.

```
            +----+
            |    |
            +----+ \
            Ari's   \
            Laptop   \
           10.1.2.3   \
            +----+     \ /                +----+
            |    |-----+----------------|    |
            +----+     / \                +----+
            Bob's     NAT                Server
            Laptop   /
           10.1.2.4 /
                   /
            +----+ /
            |    |/
            +----+
            Cliff's
            Laptop
           10.1.2.5
```
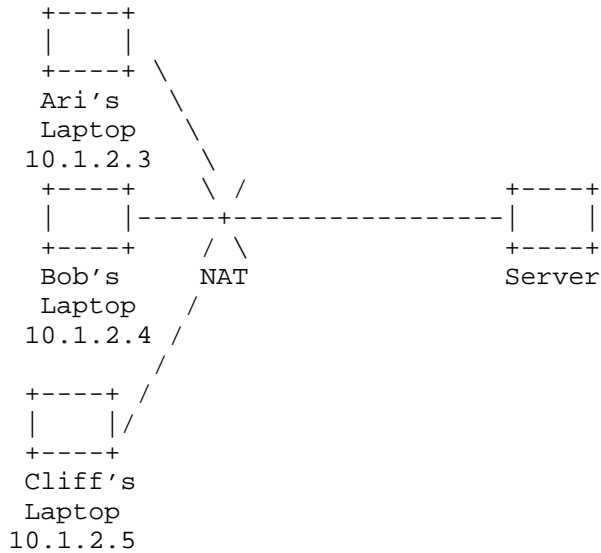
   Now, transport SAs on the server will look like this:

   To Ari: Server to NAT, <traffic desc1>, UDP encap <4500, Y>

   To Bob: Server to NAT, <traffic desc2>, UDP encap <4500, Z>

   Cliff's traffic is in the clear, so there is no SA.

   <traffic desc> is the protocol and port information.  The UDP encap
   ports are the ports used in UDP-encapsulated ESP format of section
   2.1.  Y,Z are the dynamic ports assigned by the NAT during the IKE
   negotiation.  So IKE traffic from Ari's laptop goes out on UDP
   <4500,4500>.  It reaches the server as UDP <Y,4500>, where Y is the
   dynamically assigned port.

   If the <traffic desc1> overlaps <traffic desc2>, then simple filter
   lookups may not be sufficient to determine which SA has to be used to
   send traffic.  Implementations MUST handle this situation, either by
   disallowing conflicting connections, or by other means.

Assume now that Cliff wants to connect to the server in the clear.
This is going to be difficult to configure, as the server already has
a policy (from Server to the NAT's external address) for securing
<traffic desc>.  For totally non-overlapping traffic descriptions,
this is possible.

Sample server policy could be as follows:

To Ari: Server to NAT, All UDP, secure

To Bob: Server to NAT, All TCP, secure

To Cliff: Server to NAT, ALL ICMP, clear text

Note that this policy also lets Ari and Bob send cleartext ICMP to
the server.

The server sees all clients behind the NAT as the same IP address, so
setting up different policies for the same traffic descriptor is in
principle impossible.

A problematic example of configuration on the server is as follows:

Server to NAT, TCP, secure (for Ari and Bob)

Server to NAT, TCP, clear (for Cliff)

The server cannot enforce his policy, as it is possible that
misbehaving Bob sends traffic in the clear.  This is
indistinguishable from when Cliff sends traffic in the clear.  So it
is impossible to guarantee security from some clients behind a NAT,
while allowing clear text from different clients behind the SAME NAT.
If the server's security policy allows this, however, it can do
best-effort security: If the client from behind the NAT initiates
security, his connection will be secured.  If he sends in the clear,
the server will still accept that clear text.

For security guarantees, the above problematic scenario MUST NOT be
allowed on servers.  For best effort security, this scenario MAY be
used.

Please see Appendix A.

6.  IAB Considerations

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

7.  Acknowledgments

   Thanks to Tero Kivinen and William Dixon, who contributed actively to
   this document.

   Thanks to Joern Sierwald, Tamir Zegman, Tatu Ylonen, and Santeri
   Paavolainen, who contributed to the early documents about NAT
   traversal.

8.  References

8.1.  Normative References

   [RFC0768]  Postel, J., "User Datagram Protocol", STD 6, RFC 768,
              August 1980.

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

   [RFC2401]  Kent, S. and R. Atkinson, "Security Architecture for the
              Internet Protocol", RFC 2401, November 1998.

   [RFC2406]  Kent, S. and R. Atkinson, "IP Encapsulating Security
              Payload (ESP)", RFC 2406, November 1998.

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

   [RFC3947]  Kivinen, T., "Negotiation of NAT-Traversal in the IKE",
              RFC 3947, January 2005.

8.2.  Informative References

   [RFC1122]  Braden, R., "Requirements for Internet Hosts -
              Communication Layers", STD 3, RFC 1122, October 1989.

   [RFC3193]  Patel, B., Aboba, B., Dixon, W., Zorn, G., and S. Booth,
              "Securing L2TP using IPsec", RFC 3193, November 2001.

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

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

   [IKEv2]    Kaufman, C., "Internet Key Exchange (IKEv2) Protocol",
              Work in Progress, October 2004.

Appendix A.  Clarification of Potential NAT Multiple Client Solutions

   This appendix provides clarification about potential solutions to the
   problem of multiple clients behind the same NAT simultaneously
   connecting to the same destination IP address.

   Sections 5.1 and 5.2 say that you MUST avoid this problem.  As this
   is not a matter of wire protocol, but a matter local implementation,
   the mechanisms do not belong in the protocol specification itself.
   They are instead listed in this appendix.

   Choosing an option will likely depend on the scenarios for which one
   uses/supports IPsec NAT-T.  This list is not meant to be exhaustive,
   so other solutions may exist.  We first describe the generic choices
   that solve the problem for all upper-layer protocols.

   Generic choices for ESP transport mode:

   Tr1) Implement a built-in NAT (network address translation) above
   IPsec decapsulation.

   Tr2) Implement a built-in NAPT (network address port translation)
   above IPsec decapsulation.

   Tr3) An initiator may decide not to request transport mode once NAT
   is detected and may instead request a tunnel-mode SA.  This may be a
   retry after transport mode is denied by the responder, or the
   initiator may choose to propose a tunnel SA initially.  This is no
   more difficult than knowing whether to propose transport mode or
   tunnel mode without NAT.  If for some reason the responder prefers or
   requires tunnel mode for NAT traversal, it must reject the quick mode
   SA proposal for transport mode.

   Generic choices for ESP tunnel mode:

   Tn1) Same as Tr1.

   Tn2) Same as Tr2.

   Tn3) This option is possible if an initiator can be assigned an
   address through its tunnel SA, with the responder using DHCP.  The
   initiator may initially request an internal address via the
   DHCP-IPsec method, regardless of whether it knows it is behind a NAT.
   It may re-initiate an IKE quick mode negotiation for DHCP tunnel SA
   after the responder fails the quick mode SA transport mode proposal.
   This happens either when a NAT-OA payload is sent or because it

discovers from NAT-D that the initiator is behind a NAT and its local configuration/policy will only accept a NAT connection when being assigned an address through DHCP-IPsec.

There are also implementation choices that offer limited interoperability.  Implementors should specify which applications or protocols should work if these options are selected.  Note that neither Tr4 nor Tn4, as described below, are expected to work with TCP traffic.

Limited interoperability choices for ESP transport mode:

Tr4) Implement upper-layer protocol awareness of the inbound and outbound IPsec SA so that it doesn't use the source IP and the source port as the session identifier (e.g., an L2TP session ID mapped to the IPsec SA pair that doesn't use the UDP source port or the source IP address for peer uniqueness).

Tr5) Implement application integration with IKE initiation so that it can rebind to a different source port if the IKE quick mode SA proposal is rejected by the responder; then it can repropose the new QM selector.

Limited interoperability choices for ESP tunnel mode:

Tn4) Same as Tr4.

Authors' Addresses

    Ari Huttunen
    F-Secure Corporation
    Tammasaarenkatu 7
    HELSINKI  FIN-00181
    FI

    EMail: Ari.Huttunen@F-Secure.com


    Brian Swander
    Microsoft
    One Microsoft Way
    Redmond, WA  98052
    US

    EMail: briansw@microsoft.com


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

    EMail: vvolpe@cisco.com


    Larry DiBurro
    Nortel Networks
    80 Central Street
    Boxborough, MA  01719
    US

    EMail: ldiburro@nortelnetworks.com


    Markus Stenberg
    FI

    EMail: markus.stenberg@iki.fi

Full Copyright Statement

Intellectual Property

Acknowledgement