                 IP Encapsulating Security Payload (ESP)

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 an updated version of the Encapsulating
   Security Payload (ESP) protocol, which is designed to provide a mix
   of security services in IPv4 and IPv6. ESP is used to provide
   confidentiality, data origin authentication, connectionless
   integrity, an anti-replay service (a form of partial sequence
   integrity), and limited traffic flow confidentiality. This document
   obsoletes RFC 2406 (November 1998).

Table of Contents

1. Introduction

   This document assumes that the reader is familiar with the terms and
   concepts described in the "Security Architecture for the Internet
   Protocol" [Ken-Arch], hereafter referred to as the Security
   Architecture document. In particular, the reader should be familiar
   with the definitions of security services offered by the
   Encapsulating Security Payload (ESP) and the IP Authentication Header
   (AH), the concept of Security Associations, the ways in which ESP can
   be used in conjunction with AH, and the different key management
   options available for ESP and AH.

   The keywords MUST, MUST NOT, REQUIRED, SHALL, SHALL NOT, SHOULD,
   SHOULD NOT, RECOMMENDED, MAY, and OPTIONAL, when they appear in this
   document, are to be interpreted as described in RFC 2119 [Bra97].

   The Encapsulating Security Payload (ESP) header is designed to
   provide a mix of security services in IPv4 and IPv6 [DH98]. ESP may
   be applied alone, in combination with AH [Ken-AH], or in a nested
   fashion (see the Security Architecture document [Ken-Arch]).
   Security services can be provided between a pair of communicating
   hosts, between a pair of communicating security gateways, or between
   a security gateway and a host. For more details on how to use ESP
   and AH in various network environments, see the Security Architecture
   document [Ken-Arch].

   The ESP header is inserted after the IP header and before the next
   layer protocol header (transport mode) or before an encapsulated IP
   header (tunnel mode). These modes are described in more detail
   below.

   ESP can be used to provide confidentiality, data origin
   authentication, connectionless integrity, an anti-replay service (a
   form of partial sequence integrity), and (limited) traffic flow
   confidentiality. The set of services provided depends on options
   selected at the time of Security Association (SA) establishment and
   on the location of the implementation in a network topology.

   Using encryption-only for confidentiality is allowed by ESP.
   However, it should be noted that in general, this will provide
   defense only against passive attackers. Using encryption without a
   strong integrity mechanism on top of it (either in ESP or separately
   via AH) may render the confidentiality service insecure against some
   forms of active attacks [Bel96, Kra01]. Moreover, an underlying
   integrity service, such as AH, applied before encryption does not
   necessarily protect the encryption-only confidentiality against
   active attackers [Kra01]. ESP allows encryption-only SAs because
   this may offer considerably better performance and still provide

adequate security, e.g., when higher-layer authentication/integrity protection is offered independently. However, this standard does not require ESP implementations to offer an encryption-only service.

Data origin authentication and connectionless integrity are joint services, hereafter referred to jointly as "integrity". (This term is employed because, on a per-packet basis, the computation being performed provides connectionless integrity directly; data origin authentication is provided indirectly as a result of binding the key used to verify the integrity to the identity of the IPsec peer. Typically, this binding is effected through the use of a shared, symmetric key.) Integrity-only ESP MUST be offered as a service selection option, e.g., it must be negotiable in SA management protocols and MUST be configurable via management interfaces. Integrity-only ESP is an attractive alternative to AH in many contexts, e.g., because it is faster to process and more amenable to pipelining in many implementations.

Although confidentiality and integrity can be offered independently, ESP typically will employ both services, i.e., packets will be protected with regard to confidentiality and integrity. Thus, there are three possible ESP security service combinations involving these services:

        - confidentiality-only (MAY be supported)
        - integrity only (MUST be supported)
        - confidentiality and integrity (MUST be supported)

The anti-replay service may be selected for an SA only if the integrity service is selected for that SA. The selection of this service is solely at the discretion of the receiver and thus need not be negotiated. However, to make use of the Extended Sequence Number feature in an interoperable fashion, ESP does impose a requirement on SA management protocols to be able to negotiate this feature (see Section 2.2.1 below).

The traffic flow confidentiality (TFC) service generally is effective only if ESP is employed in a fashion that conceals the ultimate source and destination addresses of correspondents, e.g., in tunnel mode between security gateways, and only if sufficient traffic flows between IPsec peers (either naturally or as a result of generation of masking traffic) to conceal the characteristics of specific, individual subscriber traffic flows. (ESP may be employed as part of a higher-layer TFC system, e.g., Onion Routing [Syverson], but such systems are outside the scope of this standard.) New TFC features present in ESP facilitate efficient generation and discarding of dummy traffic and better padding of real traffic, in a backward-compatible fashion.

   Section 7 provides a brief review of the differences between this
   document and RFC 2406.

2. Encapsulating Security Payload Packet Format

   The (outer) protocol header (IPv4, IPv6, or Extension) that
   immediately precedes the ESP header SHALL contain the value 50 in its
   Protocol (IPv4) or Next Header (IPv6, Extension) field (see IANA web
   page at http://www.iana.org/assignments/protocol-numbers). Figure 1
   illustrates the top-level format of an ESP packet. The packet begins
   with two 4-byte fields (Security Parameters Index (SPI) and Sequence
   Number). Following these fields is the Payload Data, which has
   substructure that depends on the choice of encryption algorithm and
   mode, and on the use of TFC padding, which is examined in more detail
   later. Following the Payload Data are Padding and Pad Length fields,
   and the Next Header field. The optional Integrity Check Value (ICV)
   field completes the packet.

```
 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
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ----
|               Security Parameters Index (SPI)                 | ^Int.
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|                      Sequence Number                          | |ered
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | ----
|                    Payload Data* (variable)                   | |   ^
~                                                               ~ |   |
|                                                               | |Conf.
+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |Cov-
|               |         Padding (0-255 bytes)                 | |ered*
+-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ |   |
|               |               |  Pad Length   | Next Header   | v   v
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ ------
|         Integrity Check Value-ICV   (variable)                |
~                                                               ~
|                                                               |
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```

                Figure 1. Top-Level Format of an ESP Packet

     * If included in the Payload field, cryptographic synchronization
       data, e.g., an Initialization Vector (IV, see Section 2.3),
       usually is not encrypted per se, although it often is referred
       to as being part of the ciphertext.

The (transmitted) ESP trailer consists of the Padding, Pad Length, and Next Header fields. Additional, implicit ESP trailer data (which is not transmitted) is included in the integrity computation, as described below.

If the integrity service is selected, the integrity computation encompasses the SPI, Sequence Number, Payload Data, and the ESP trailer (explicit and implicit).

If the confidentiality service is selected, the ciphertext consists of the Payload Data (except for any cryptographic synchronization data that may be included) and the (explicit) ESP trailer.

As noted above, the Payload Data may have substructure. An encryption algorithm that requires an explicit Initialization Vector (IV), e.g., Cipher Block Chaining (CBC) mode, often prefixes the Payload Data to be protected with that value. Some algorithm modes combine encryption and integrity into a single operation; this document refers to such algorithm modes as "combined mode algorithms". Accommodation of combined mode algorithms requires that the algorithm explicitly describe the payload substructure used to convey the integrity data.

Some combined mode algorithms provide integrity only for data that is encrypted, whereas others can provide integrity for some additional data that is not encrypted for transmission. Because the SPI and Sequence Number fields require integrity as part of the integrity service, and they are not encrypted, it is necessary to ensure that they are afforded integrity whenever the service is selected, regardless of the style of combined algorithm mode employed.

When any combined mode algorithm is employed, the algorithm itself is expected to return both decrypted plaintext and a pass/fail indication for the integrity check. For combined mode algorithms, the ICV that would normally appear at the end of the ESP packet (when integrity is selected) may be omitted. When the ICV is omitted and integrity is selected, it is the responsibility of the combined mode algorithm to encode within the Payload Data an ICV-equivalent means of verifying the integrity of the packet.

If a combined mode algorithm offers integrity only to data that is encrypted, it will be necessary to replicate the SPI and Sequence Number as part of the Payload Data.

Finally, a new provision is made to insert padding for traffic flow confidentiality after the Payload Data and before the ESP trailer. Figure 2 illustrates this substructure for Payload Data. (Note: This

diagram shows bits-on-the-wire. So even if extended sequence numbers
are being used, only 32 bits of the Sequence Number will be
transmitted (see Section 2.2.1).)

```
   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
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |               Security Parameters Index (SPI)                 |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                      Sequence Number                          |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---
  |                    IV (optional)                             | ^ p
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | a
  |                  Rest of Payload Data  (variable)            | | y
  ~                                                              ~ | l
  |                                                              | | o
  +               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+ | a
  |               |         TFC Padding * (optional, variable)   | v d
  +-+-+-+-+-+-+-+-+               +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+---
  |                               |         Padding (0-255 bytes) |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+     +-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |                               | Pad Length   | Next Header    |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
  |         Integrity Check Value-ICV   (variable)               |
  ~                                                              ~
  |                                                              |
  +-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
```
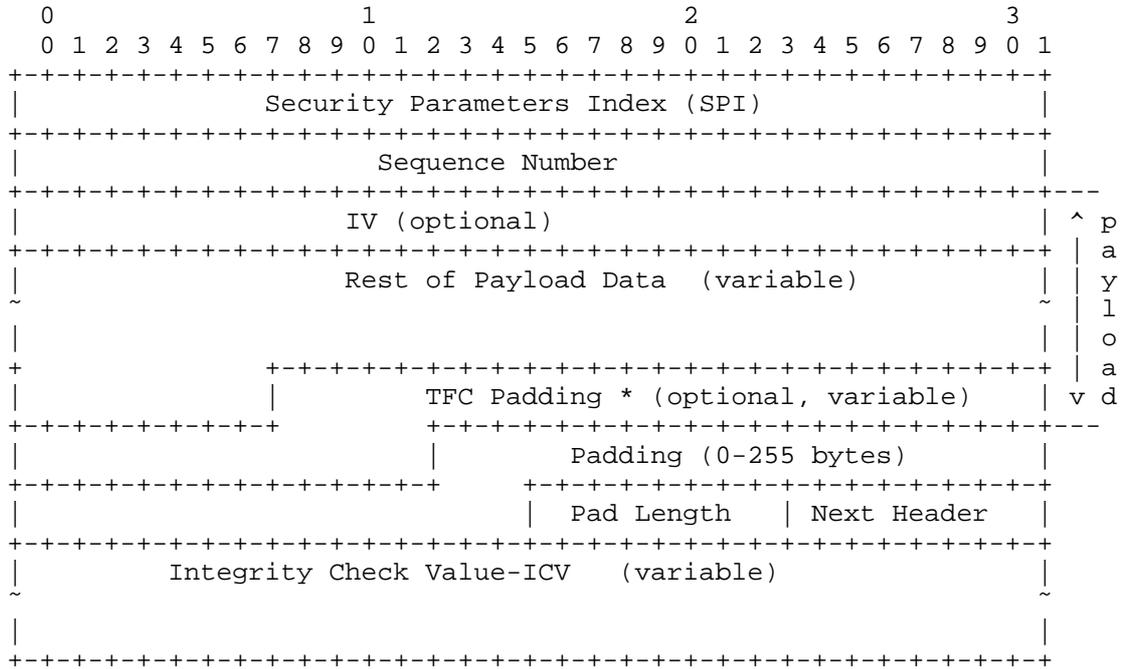
                 Figure 2. Substructure of Payload Data

        * If tunnel mode is being used, then the IPsec implementation
          can add Traffic Flow Confidentiality (TFC) padding (see
          Section 2.4) after the Payload Data and before the Padding
          (0-255 bytes) field.

If a combined algorithm mode is employed, the explicit ICV shown in
Figures 1 and 2 may be omitted (see Section 3.3.2.2 below). Because
algorithms and modes are fixed when an SA is established, the
detailed format of ESP packets for a given SA (including the Payload
Data substructure) is fixed, for all traffic on the SA.

The tables below refer to the fields in the preceding figures and
illustrate how several categories of algorithmic options, each with a
different processing model, affect the fields noted above. The
processing details are described in later sections.

Table 1. Separate Encryption and Integrity Algorithms

```
                                   What    What    What
                        # of     Requ'd  Encrypt  Integ    is
                        bytes      [1]   Covers   Covers   Xmtd
                        ------    ------  ------   ------   ------
    SPI                   4         M               Y      plain
    Seq# (low-order bits) 4         M               Y      plain         p
                                                                  ------  a
    IV                  variable    O               Y      plain    |    y
    IP datagram [2]     variable  M or D    Y       Y      cipher[3]|-l
    TFC padding [4]     variable    O       Y       Y      cipher[3]| o
                                                                  ------  a
    Padding             0-255       M       Y       Y      cipher[3]  d
    Pad Length            1         M       Y       Y      cipher[3]
    Next Header           1         M       Y       Y      cipher[3]
    Seq# (high-order bits) 4     if ESN [5]         Y      not xmtd
    ICV Padding         variable if need           Y      not xmtd
    ICV                 variable   M [6]                   plain
```

            [1] M = mandatory; O = optional; D = dummy
            [2] If tunnel mode -> IP datagram
                If transport mode -> next header and data
            [3] ciphertext if encryption has been selected
            [4] Can be used only if payload specifies its "real" length
            [5] See section 2.2.1
            [6] mandatory if a separate integrity algorithm is used

Table 2. Combined Mode Algorithms

	# of bytes	Requ'd [1]	What Encrypt Covers	What Integ Covers	What is Xmtd	
SPI	4	M			plain	
Seq# (low-order bits)	4	M			plain	p
					---	a
IV	variable	O		Y	plain |	y
IP datagram [2]	variable	M or D	Y	Y	cipher |	-l
TFC padding [3]	variable	O	Y	Y	cipher |	o
					---	a
Padding	0-255	M	Y	Y	cipher	d
Pad Length	1	M	Y	Y	cipher	
Next Header	1	M	Y	Y	cipher	
Seq# (high-order bits)	4	if ESN [4]		Y	[5]	
ICV Padding	variable	if need		Y	[5]	
ICV	variable	O [6]			plain	

```
        [1] M = mandatory; O = optional; D = dummy
        [2] If tunnel mode -> IP datagram
            If transport mode -> next header and data
        [3] Can be used only if payload specifies its "real" length
        [4] See Section 2.2.1
        [5] The algorithm choices determines whether these are
            transmitted, but in either case, the result is invisible
            to ESP
        [6] The algorithm spec determines whether this field is
            present
```

The following subsections describe the fields in the header format.
"Optional" means that the field is omitted if the option is not
selected, i.e., it is present in neither the packet as transmitted
nor as formatted for computation of an ICV (see Section 2.7).
Whether or not an option is selected is determined as part of
Security Association (SA) establishment. Thus, the format of ESP
packets for a given SA is fixed, for the duration of the SA. In
contrast, "mandatory" fields are always present in the ESP packet
format, for all SAs.

Note: All of the cryptographic algorithms used in IPsec expect their
input in canonical network byte order (see Appendix of RFC 791
[Pos81]) and generate their output in canonical network byte order.
IP packets are also transmitted in network byte order.

ESP does not contain a version number, therefore if there are
concerns about backward compatibility, they MUST be addressed by
using a signaling mechanism between the two IPsec peers to ensure
compatible versions of ESP (e.g., Internet Key Exchange (IKEv2)
[Kau05]) or an out-of-band configuration mechanism.

2.1. Security Parameters Index (SPI)

The SPI is an arbitrary 32-bit value that is used by a receiver to
identify the SA to which an incoming packet is bound. The SPI field
is mandatory.

For a unicast SA, the SPI can be used by itself to specify an SA, or
it may be used in conjunction with the IPsec protocol type (in this
case ESP). Because the SPI value is generated by the receiver for a
unicast SA, whether the value is sufficient to identify an SA by
itself or whether it must be used in conjunction with the IPsec
protocol value is a local matter. This mechanism for mapping inbound
traffic to unicast SAs MUST be supported by all ESP implementations.

If an IPsec implementation supports multicast, then it MUST support
multicast SAs using the algorithm below for mapping inbound IPsec
datagrams to SAs. Implementations that support only unicast traffic
need not implement this de-multiplexing algorithm.

In many secure multicast architectures (e.g., [RFC3740]), a central
Group Controller/Key Server unilaterally assigns the group security
association's SPI. This SPI assignment is not negotiated or
coordinated with the key management (e.g., IKE) subsystems that
reside in the individual end systems that comprise the group.
Consequently, it is possible that a group security association and a
unicast security association can simultaneously use the same SPI. A
multicast-capable IPsec implementation MUST correctly de-multiplex
inbound traffic even in the context of SPI collisions.

Each entry in the Security Association Database (SAD) [Ken-Arch] must
indicate whether the SA lookup makes use of the destination, or
destination and source, IP addresses, in addition to the SPI. For
multicast SAs, the protocol field is not employed for SA lookups.
For each inbound, IPsec-protected packet, an implementation must
conduct its search of the SAD such that it finds the entry that
matches the "longest" SA identifier. In this context, if two or more
SAD entries match based on the SPI value, then the entry that also
matches based on destination, or destination and source, address
comparison (as indicated in the SAD entry) is the "longest" match.
This implies a logical ordering of the SAD search as follows:

1. Search the SAD for a match on {SPI, destination address,
   source address}. If an SAD entry matches, then process the
   inbound ESP packet with that matching SAD entry. Otherwise,
   proceed to step 2.

2. Search the SAD for a match on {SPI, destination address}.
   If the SAD entry matches, then process the inbound ESP
   packet with that matching SAD entry. Otherwise, proceed to
   step 3.

3. Search the SAD for a match on only {SPI} if the receiver has
   chosen to maintain a single SPI space for AH and ESP, or on
   {SPI, protocol} otherwise. If an SAD entry matches, then
   process the inbound ESP packet with that matching SAD entry.
   Otherwise, discard the packet and log an auditable event.

In practice, an implementation MAY choose any method to accelerate
this search, although its externally visible behavior MUST be
functionally equivalent to having searched the SAD in the above
order. For example, a software-based implementation could index into
a hash table by the SPI. The SAD entries in each hash table bucket's
linked list are kept sorted to have those SAD entries with the
longest SA identifiers first in that linked list. Those SAD entries
having the shortest SA identifiers are sorted so that they are the
last entries in the linked list. A hardware-based implementation may
be able to effect the longest match search intrinsically, using
commonly available Ternary Content-Addressable Memory (TCAM)
features.

The indication of whether source and destination address matching is
required to map inbound IPsec traffic to SAs MUST be set either as a
side effect of manual SA configuration or via negotiation using an SA
management protocol, e.g., IKE or Group Domain of Interpretation
(GDOI) [RFC3547]. Typically, Source-Specific Multicast (SSM) [HC03]
groups use a 3-tuple SA identifier composed of an SPI, a destination
multicast address, and source address. An Any-Source Multicast group
SA requires only an SPI and a destination multicast address as an
identifier.

The set of SPI values in the range 1 through 255 are reserved by the
Internet Assigned Numbers Authority (IANA) for future use; a reserved
SPI value will not normally be assigned by IANA unless the use of the
assigned SPI value is specified in an RFC. The SPI value of zero (0)
is reserved for local, implementation-specific use and MUST NOT be
sent on the wire. (For example, a key management implementation
might use the zero SPI value to mean "No Security Association Exists"

during the period when the IPsec implementation has requested that
its key management entity establish a new SA, but the SA has not yet
been established.)

2.2. Sequence Number

This unsigned 32-bit field contains a counter value that increases by
one for each packet sent, i.e., a per-SA packet sequence number. For
a unicast SA or a single-sender multicast SA, the sender MUST
increment this field for every transmitted packet. Sharing an SA
among multiple senders is permitted, though generally not
recommended. ESP provides no means of synchronizing packet counters
among multiple senders or meaningfully managing a receiver packet
counter and window in the context of multiple senders. Thus, for a
multi-sender SA, the anti-replay features of ESP are not available
(see Sections 3.3.3 and 3.4.3.)

The field is mandatory and MUST always be present even if the
receiver does not elect to enable the anti-replay service for a
specific SA. Processing of the Sequence Number field is at the
discretion of the receiver, but all ESP implementations MUST be
capable of performing the processing described in Sections 3.3.3 and
3.4.3. Thus, the sender MUST always transmit this field, but the
receiver need not act upon it (see the discussion of Sequence Number
Verification in the "Inbound Packet Processing" section (3.4.3)
below).

The sender's counter and the receiver's counter are initialized to 0
when an SA is established. (The first packet sent using a given SA
will have a sequence number of 1; see Section 3.3.3 for more details
on how the sequence number is generated.) If anti-replay is enabled
(the default), the transmitted sequence number must never be allowed
to cycle. Thus, the sender's counter and the receiver's counter MUST
be reset (by establishing a new SA and thus a new key) prior to the
transmission of the 2^{32}nd packet on an SA.

2.2.1. Extended (64-bit) Sequence Number

To support high-speed IPsec implementations, Extended Sequence
Numbers (ESNs) SHOULD be implemented, as an extension to the current,
32-bit sequence number field. Use of an ESN MUST be negotiated by an
SA management protocol. Note that in IKEv2, this negotiation is
implicit; the default is ESN unless 32-bit sequence numbers are
explicitly negotiated. (The ESN feature is applicable to multicast
as well as unicast SAs.)

The ESN facility allows use of a 64-bit sequence number for an SA.
(See Appendix A, "Extended (64-bit) Sequence Numbers", for details.)
Only the low-order 32 bits of the sequence number are transmitted in
the plaintext ESP header of each packet, thus minimizing packet
overhead. The high-order 32 bits are maintained as part of the
sequence number counter by both transmitter and receiver and are
included in the computation of the ICV (if the integrity service is
selected). If a separate integrity algorithm is employed, the high
order bits are included in the implicit ESP trailer, but are not
transmitted, analogous to integrity algorithm padding bits. If a
combined mode algorithm is employed, the algorithm choice determines
whether the high-order ESN bits are transmitted or are included
implicitly in the computation. See Section 3.3.2.2 for processing
details.

2.3. Payload Data

   Payload Data is a variable-length field containing data (from the
   original IP packet) described by the Next Header field. The Payload
   Data field is mandatory and is an integral number of bytes in length.
   If the algorithm used to encrypt the payload requires cryptographic
   synchronization data, e.g., an Initialization Vector (IV), then this
   data is carried explicitly in the Payload field, but it is not called
   out as a separate field in ESP, i.e., the transmission of an explicit
   IV is invisible to ESP. (See Figure 2.) Any encryption algorithm
   that requires such explicit, per-packet synchronization data MUST
   indicate the length, any structure for such data, and the location of
   this data as part of an RFC specifying how the algorithm is used with
   ESP. (Typically, the IV immediately precedes the ciphertext. See
   Figure 2.) If such synchronization data is implicit, the algorithm
   for deriving the data MUST be part of the algorithm definition RFC.
   (If included in the Payload field, cryptographic synchronization
   data, e.g., an Initialization Vector (IV), usually is not encrypted
   per se (see Tables 1 and 2), although it sometimes is referred to as
   being part of the ciphertext.)

   Note that the beginning of the next layer protocol header MUST be
   aligned relative to the beginning of the ESP header as follows. For
   IPv4, this alignment is a multiple of 4 bytes. For IPv6, the
   alignment is a multiple of 8 bytes.

   With regard to ensuring the alignment of the (real) ciphertext in the
   presence of an IV, note the following:

        o For some IV-based modes of operation, the receiver treats
          the IV as the start of the ciphertext, feeding it into the
          algorithm directly. In these modes, alignment of the start
          of the (real) ciphertext is not an issue at the receiver.

          o In some cases, the receiver reads the IV in separately from
            the ciphertext. In these cases, the algorithm specification
            MUST address how alignment of the (real) ciphertext is to be
            achieved.

2.4. Padding (for Encryption)

   Two primary factors require or motivate use of the Padding field.

      o If an encryption algorithm is employed that requires the
        plaintext to be a multiple of some number of bytes, e.g.,
        the block size of a block cipher, the Padding field is used
        to fill the plaintext (consisting of the Payload Data,
        Padding, Pad Length, and Next Header fields) to the size
        required by the algorithm.

      o Padding also may be required, irrespective of encryption
        algorithm requirements, to ensure that the resulting
        ciphertext terminates on a 4-byte boundary. Specifically,
        the Pad Length and Next Header fields must be right aligned
        within a 4-byte word, as illustrated in the ESP packet
        format figures above, to ensure that the ICV field (if
        present) is aligned on a 4-byte boundary.

   Padding beyond that required for the algorithm or alignment reasons
   cited above could be used to conceal the actual length of the
   payload, in support of TFC. However, the Padding field described is
   too limited to be effective for TFC and thus should not be used for
   that purpose. Instead, the separate mechanism described below (see
   Section 2.7) should be used when TFC is required.

   The sender MAY add 0 to 255 bytes of padding. Inclusion of the
   Padding field in an ESP packet is optional, subject to the
   requirements noted above, but all implementations MUST support
   generation and consumption of padding.

      o For the purpose of ensuring that the bits to be encrypted
        are a multiple of the algorithm's block size (first bullet
        above), the padding computation applies to the Payload Data
        exclusive of any IV, but including the ESP trailer
        fields. If a combined algorithm mode requires transmission
        of the SPI and Sequence Number to effect integrity, e.g.,
        replication of the SPI and Sequence Number in the Payload
        Data, then the replicated versions of these data items, and
        any associated, ICV-equivalent data, are included in the
        computation of the pad length. (If the ESN option is

        selected, the high-order 32 bits of the ESN also would enter
        into the computation, if the combined mode algorithm
        requires their transmission for integrity.)

      o For the purposes of ensuring that the ICV is aligned on a
        4-byte boundary (second bullet above), the padding
        computation applies to the Payload Data inclusive of the IV,
        the Pad Length, and Next Header fields. If a combined mode
        algorithm is used, any replicated data and ICV-equivalent
        data are included in the Payload Data covered by the padding
        computation.

   If Padding bytes are needed but the encryption algorithm does not
   specify the padding contents, then the following default processing
   MUST be used. The Padding bytes are initialized with a series of
   (unsigned, 1-byte) integer values. The first padding byte appended
   to the plaintext is numbered 1, with subsequent padding bytes making
   up a monotonically increasing sequence: 1, 2, 3, ... When this
   padding scheme is employed, the receiver SHOULD inspect the Padding
   field. (This scheme was selected because of its relative simplicity,
   ease of implementation in hardware, and because it offers limited
   protection against certain forms of "cut and paste" attacks in the
   absence of other integrity measures, if the receiver checks the
   padding values upon decryption.)

   If an encryption or combined mode algorithm imposes constraints on
   the values of the bytes used for padding, they MUST be specified by
   the RFC defining how the algorithm is employed with ESP. If the
   algorithm requires checking of the values of the bytes used for
   padding, this too MUST be specified in that RFC.

2.5. Pad Length

   The Pad Length field indicates the number of pad bytes immediately
   preceding it in the Padding field. The range of valid values is 0 to
   255, where a value of zero indicates that no Padding bytes are
   present. As noted above, this does not include any TFC padding
   bytes. The Pad Length field is mandatory.

2.6. Next Header

   The Next Header is a mandatory, 8-bit field that identifies the type
   of data contained in the Payload Data field, e.g., an IPv4 or IPv6
   packet, or a next layer header and data. The value of this field is
   chosen from the set of IP Protocol Numbers defined on the web page of
   the IANA, e.g., a value of 4 indicates IPv4, a value of 41 indicates
   IPv6, and a value of 6 indicates TCP.

   To facilitate the rapid generation and discarding of the padding
   traffic in support of traffic flow confidentiality (see Section 2.4),
   the protocol value 59 (which means "no next header") MUST be used to
   designate a "dummy" packet. A transmitter MUST be capable of
   generating dummy packets marked with this value in the next protocol
   field, and a receiver MUST be prepared to discard such packets,
   without indicating an error. All other ESP header and trailer fields
   (SPI, Sequence Number, Padding, Pad Length, Next Header, and ICV)
   MUST be present in dummy packets, but the plaintext portion of the
   payload, other than this Next Header field, need not be well-formed,
   e.g., the rest of the Payload Data may consist of only random bytes.
   Dummy packets are discarded without prejudice.

   Implementations SHOULD provide local management controls to enable
   the use of this capability on a per-SA basis. The controls should
   allow the user to specify if this feature is to be used and also
   provide parametric controls; for example, the controls might allow an
   administrator to generate random-length or fixed-length dummy
   packets.

   DISCUSSION: Dummy packets can be inserted at random intervals to mask
   the absence of actual traffic. One can also "shape" the actual
   traffic to match some distribution to which dummy traffic is added as
   dictated by the distribution parameters. As with the packet length
   padding facility for Traffic Flow Security (TFS), the most secure
   approach would be to generate dummy packets at whatever rate is
   needed to maintain a constant rate on an SA. If packets are all the
   same size, then the SA presents the appearance of a constant bit rate
   data stream, analogous to what a link crypto would offer at layer 1
   or 2. However, this is unlikely to be practical in many contexts,
   e.g., when there are multiple SAs active, because it would imply
   reducing the allowed bandwidth for a site, based on the number of
   SAs, and that would undermine the benefits of packet switching.
   Implementations SHOULD provide controls to enable local
   administrators to manage the generation of dummy packets for TFC
   purposes.

2.7. Traffic Flow Confidentiality (TFC) Padding

   As noted above, the Padding field is limited to 255 bytes in length.
   This generally will not be adequate to hide traffic characteristics
   relative to traffic flow confidentiality requirements. An optional
   field, within the payload data, is provided specifically to address
   the TFC requirement.

   An IPsec implementation SHOULD be capable of padding traffic by
   adding bytes after the end of the Payload Data, prior to the
   beginning of the Padding field. However, this padding (hereafter
   referred to as TFC padding) can be added only if the Payload Data
   field contains a specification of the length of the IP datagram.
   This is always true in tunnel mode, and may be true in transport mode
   depending on whether the next layer protocol (e.g., IP, UDP, ICMP)
   contains explicit length information. This length information will
   enable the receiver to discard the TFC padding, because the true
   length of the Payload Data will be known. (ESP trailer fields are
   located by counting back from the end of the ESP packet.)
   Accordingly, if TFC padding is added, the field containing the
   specification of the length of the IP datagram MUST NOT be modified
   to reflect this padding. No requirements for the value of this
   padding are established by this standard.

   In principle, existing IPsec implementations could have made use of
   this capability previously, in a transparent fashion. However,
   because receivers may not have been prepared to deal with this
   padding, the SA management protocol MUST negotiate this service prior
   to a transmitter employing it, to ensure backward compatibility.
   Combined with the convention described in Section 2.6 above, about
   the use of protocol ID 59, an ESP implementation is capable of
   generating dummy and real packets that exhibit much greater length
   variability, in support of TFC.

   Implementations SHOULD provide local management controls to enable
   the use of this capability on a per-SA basis. The controls should
   allow the user to specify if this feature is to be used and also
   provide parametric controls for the feature.

2.8. Integrity Check Value (ICV)

   The Integrity Check Value is a variable-length field computed over
   the ESP header, Payload, and ESP trailer fields. Implicit ESP
   trailer fields (integrity padding and high-order ESN bits, if
   applicable) are included in the ICV computation. The ICV field is
   optional. It is present only if the integrity service is selected
   and is provided by either a separate integrity algorithm or a
   combined mode algorithm that uses an ICV. The length of the field is

specified by the integrity algorithm selected and associated with the
SA. The integrity algorithm specification MUST specify the length of
the ICV and the comparison rules and processing steps for validation.
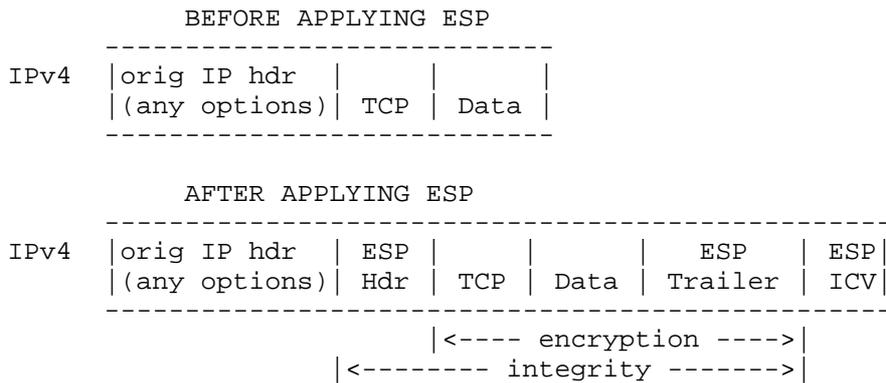
3. Encapsulating Security Protocol Processing

3.1. ESP Header Location

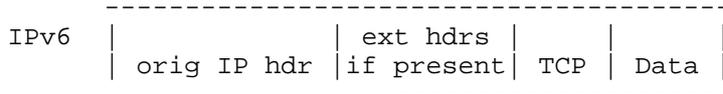   ESP may be employed in two ways: transport mode or tunnel mode.

3.1.1. Transport Mode Processing

   In transport mode, ESP is inserted after the IP header and before a
   next layer protocol, e.g., TCP, UDP, ICMP, etc. In the context of
   IPv4, this translates to placing ESP after the IP header (and any
   options that it contains), but before the next layer protocol. (If
   AH is also applied to a packet, it is applied to the ESP header,
   Payload, ESP trailer, and ICV, if present.) (Note that the term
   "transport" mode should not be misconstrued as restricting its use to
   TCP and UDP.) The following diagram illustrates ESP transport mode
   positioning for a typical IPv4 packet, on a "before and after" basis.
   (This and subsequent diagrams in this section show the ICV field, the
   presence of which is a function of the security services and the
   algorithm/mode selected.)

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

                 AFTER APPLYING ESP
            -------------------------------------------------
     IPv4  |orig IP hdr  | ESP |     |      |   ESP   | ESP|
           |(any options)| Hdr | TCP | Data | Trailer | ICV|
            -------------------------------------------------
                         |<---- encryption ---->|
                         |<-------- integrity ------->|
```
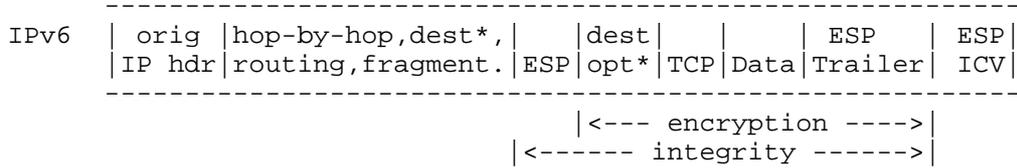
   In the IPv6 context, ESP is viewed as an end-to-end payload, and thus
   should appear after hop-by-hop, routing, and fragmentation extension
   headers. Destination options extension header(s) could appear
   before, after, or both before and after the ESP header depending on
   the semantics desired. However, because ESP protects only fields
   after the ESP header, it generally will be desirable to place the
   destination options header(s) after the ESP header. The following
   diagram illustrates ESP transport mode positioning for a typical IPv6
   packet.

```
                    BEFORE APPLYING ESP
                ----------------------------------------
        IPv6   |               | ext hdrs |     |      |
               | orig IP hdr  |if present| TCP | Data |
                ----------------------------------------

                    AFTER APPLYING ESP
                ----------------------------------------------------------
        IPv6   | orig  |hop-by-hop,dest*,|   |dest|   |    | ESP   | ESP|
               |IP hdr |routing,fragment.|ESP|opt*|TCP|Data|Trailer| ICV|
                ----------------------------------------------------------
                                          |<--- encryption ---->|
                                        |<------ integrity ------>|

             * = if present, could be before ESP, after ESP, or both
```

   Note that in transport mode, for "bump-in-the-stack" or "bump-in-
   the-wire" implementations, as defined in the Security Architecture
   document, inbound and outbound IP fragments may require an IPsec
   implementation to perform extra IP reassembly/fragmentation in order
   to both conform to this specification and provide transparent IPsec
   support. Special care is required to perform such operations within
   these implementations when multiple interfaces are in use.

3.1.2. Tunnel Mode Processing

   In tunnel mode, the "inner" IP header carries the ultimate (IP)
   source and destination addresses, while an "outer" IP header contains
   the addresses of the IPsec "peers", e.g., addresses of security
   gateways. Mixed inner and outer IP versions are allowed, i.e., IPv6
   over IPv4 and IPv4 over IPv6. In tunnel mode, ESP protects the
   entire inner IP packet, including the entire inner IP header. The
   position of ESP in tunnel mode, relative to the outer IP header, is
   the same as for ESP in transport mode. The following diagram
   illustrates ESP tunnel mode positioning for typical IPv4 and IPv6
   packets.

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

                  AFTER APPLYING ESP
           -------------------------------------------------------
   IPv4   | new IP hdr* |     |     orig IP hdr*  |   |    | ESP   | ESP|
          |(any options)| ESP |  (any options) |TCP|Data|Trailer| ICV|
           -------------------------------------------------------
                                 |<-------- encryption --------->|
                         |<------------ integrity ------------>|


                  BEFORE APPLYING ESP
           ---------------------------------------
   IPv6   |            | ext hdrs |     |     |
          | orig IP hdr |if present| TCP | Data |
           ---------------------------------------

                  AFTER APPLYING ESP
           -------------------------------------------------------
   IPv6   | new* |new ext |    | orig*|orig ext |   |    | ESP   | ESP|
          |IP hdr| hdrs*  |ESP|IP hdr| hdrs *  |TCP|Data|Trailer| ICV|
           -------------------------------------------------------
                              |<-------- encryption ---------->|
                       |<----------- integrity ------------>|
```

    * = if present, construction of outer IP hdr/extensions and
        modification of inner IP hdr/extensions is discussed in
        the Security Architecture document.

3.2. Algorithms

   The mandatory-to-implement algorithms for use with ESP are described
   in a separate RFC, to facilitate updating the algorithm requirements
   independently from the protocol per se. Additional algorithms,
   beyond those mandated for ESP, MAY be supported. Note that although
   both confidentiality and integrity are optional, at least one of
   these services MUST be selected, hence both algorithms MUST NOT be
   simultaneously NULL.

3.2.1. Encryption Algorithms

   The encryption algorithm employed to protect an ESP packet is
   specified by the SA via which the packet is transmitted/received.
   Because IP packets may arrive out of order, and not all packets may
   arrive (packet loss), each packet must carry any data required to
   allow the receiver to establish cryptographic synchronization for
   decryption. This data may be carried explicitly in the payload
   field, e.g., as an IV (as described above), or the data may be
   derived from the plaintext portions of the (outer IP or ESP) packet
   header. (Note that if plaintext header information is used to derive
   an IV, that information may become security critical and thus the
   protection boundary associated with the encryption process may grow.
   For example, if one were to use the ESP Sequence Number to derive an
   IV, the Sequence Number generation logic (hardware or software) would
   have to be evaluated as part of the encryption algorithm
   implementation. In the case of FIPS 140-2 [NIST01], this could
   significantly extend the scope of a cryptographic module evaluation.)
   Because ESP makes provision for padding of the plaintext, encryption
   algorithms employed with ESP may exhibit either block or stream mode
   characteristics. Note that because encryption (confidentiality) MAY
   be an optional service (e.g., integrity-only ESP), this algorithm MAY
   be "NULL" [Ken-Arch].

   To allow an ESP implementation to compute the encryption padding
   required by a block mode encryption algorithm, and to determine the
   MTU impact of the algorithm, the RFC for each encryption algorithm
   used with ESP must specify the padding modulus for the algorithm.

3.2.2. Integrity Algorithms

   The integrity algorithm employed for the ICV computation is specified
   by the SA via which the packet is transmitted/received. As was the
   case for encryption algorithms, any integrity algorithm employed with
   ESP must make provisions to permit processing of packets that arrive
   out of order and to accommodate packet loss. The same admonition
   noted above applies to use of any plaintext data to facilitate
   receiver synchronization of integrity algorithms. Note that because
   the integrity service MAY be optional, this algorithm may be "NULL".

   To allow an ESP implementation to compute any implicit integrity
   algorithm padding required, the RFC for each algorithm used with ESP
   must specify the padding modulus for the algorithm.

3.2.3. Combined Mode Algorithms

   If a combined mode algorithm is employed, both confidentiality and
   integrity services are provided. As was the case for encryption
   algorithms, a combined mode algorithm must make provisions for per-
   packet cryptographic synchronization, to permit decryption of packets
   that arrive out of order and to accommodate packet loss. The means
   by which a combined mode algorithm provides integrity for the
   payload, and for the SPI and (Extended) Sequence Number fields, may
   vary for different algorithm choices. In order to provide a uniform,
   algorithm-independent approach to invocation of combined mode
   algorithms, no payload substructure is defined. For example, the SPI
   and Sequence Number fields might be replicated within the ciphertext
   envelope and an ICV may be appended to the ESP trailer. None of
   these details should be observable externally.

   To allow an ESP implementation to determine the MTU impact of a
   combined mode algorithm, the RFC for each algorithm used with ESP
   must specify a (simple) formula that yields encrypted payload size,
   as a function of the plaintext payload and sequence number sizes.

3.3. Outbound Packet Processing

   In transport mode, the sender encapsulates the next layer protocol
   information between the ESP header and the ESP trailer fields, and
   retains the specified IP header (and any IP extension headers in the
   IPv6 context). In tunnel mode, the outer and inner IP
   header/extensions can be interrelated in a variety of ways. The
   construction of the outer IP header/extensions during the
   encapsulation process is described in the Security Architecture
   document.

3.3.1. Security Association Lookup

   ESP is applied to an outbound packet only after an IPsec
   implementation determines that the packet is associated with an SA
   that calls for ESP processing. The process of determining what, if
   any, IPsec processing is applied to outbound traffic is described in
   the Security Architecture document.

3.3.2. Packet Encryption and Integrity Check Value (ICV) Calculation

   In this section, we speak in terms of encryption always being applied
   because of the formatting implications. This is done with the
   understanding that "no confidentiality" is offered by using the NULL
   encryption algorithm (RFC 2410). There are several algorithmic
   options.

3.3.2.1. Separate Confidentiality and Integrity Algorithms

   If separate confidentiality and integrity algorithms are employed,
   the Sender proceeds as follows:

        1. Encapsulate (into the ESP Payload field):
                - for transport mode -- just the original next layer
                  protocol information.
                - for tunnel mode -- the entire original IP datagram.

        2. Add any necessary padding -- Optional TFC padding and
           (encryption) Padding

        3. Encrypt the result using the key, encryption algorithm,
           and algorithm mode specified for the SA and using any
           required cryptographic synchronization data.
                - If explicit cryptographic synchronization data,
                  e.g., an IV, is indicated, it is input to the
                  encryption algorithm per the algorithm specification
                  and placed in the Payload field.
                - If implicit cryptographic synchronization data is
                  employed, it is constructed and input to the
                  encryption algorithm as per the algorithm
                  specification.
                - If integrity is selected, encryption is performed
                  first, before the integrity algorithm is applied, and
                  the encryption does not encompass the ICV field.
                  This order of processing facilitates rapid detection
                  and rejection of replayed or bogus packets by the
                  receiver, prior to decrypting the packet, hence
                  potentially reducing the impact of denial of service
                  (DoS) attacks. It also allows for the possibility of
                  parallel processing of packets at the receiver, i.e.,
                  decryption can take place in parallel with integrity
                  checking. Note that because the ICV is not protected
                  by encryption, a keyed integrity algorithm must be
                  employed to compute the ICV.

        4. Compute the ICV over the ESP packet minus the ICV field.
           Thus, the ICV computation encompasses the SPI, Sequence
           Number, Payload Data, Padding (if present), Pad Length, and
           Next Header. (Note that the last 4 fields will be in
           ciphertext form, because encryption is performed first.) If
           the ESN option is enabled for the SA, the high-order 32
           bits of the sequence number are appended after the Next
           Header field for purposes of this computation, but are not
           transmitted.

For some integrity algorithms, the byte string over which the ICV computation is performed must be a multiple of a block size specified by the algorithm. If the length of ESP packet (as described above) does not match the block size requirements for the algorithm, implicit padding MUST be appended to the end of the ESP packet. (This padding is added after the Next Header field, or after the high-order 32 bits of the sequence number, if ESN is selected.) The block size (and hence the length of the padding) is specified by the integrity algorithm specification. This padding is not transmitted with the packet. The document that defines an integrity algorithm MUST be consulted to determine if implicit padding is required as described above. If the document does not specify an answer to this question, then the default is to assume that implicit padding is required (as needed to match the packet length to the algorithm's block size.) If padding bytes are needed but the algorithm does not specify the padding contents, then the padding octets MUST have a value of zero.

3.3.2.2. Combined Confidentiality and Integrity Algorithms

   If a combined confidentiality/integrity algorithm is employed, the
   Sender proceeds as follows:

         1. Encapsulate into the ESP Payload Data field:
               - for transport mode -- just the original next layer
                 protocol information.
               - for tunnel mode -- the entire original IP datagram.

         2. Add any necessary padding -- includes optional TFC padding
            and (encryption) Padding.

         3. Encrypt and integrity protect the result using the key
            and combined mode algorithm specified for the SA and using
            any required cryptographic synchronization data.
               - If explicit cryptographic synchronization data,
                 e.g., an IV, is indicated, it is input to the
                 combined mode algorithm per the algorithm
                 specification and placed in the Payload field.
               - If implicit cryptographic synchronization data is
                 employed, it is constructed and input to the
                 encryption algorithm as per the algorithm
                 specification.
               - The Sequence Number (or Extended Sequence Number, as
                 appropriate) and the SPI are inputs to the
                 algorithm, as they must be included in the integrity
                 check computation. The means by which these values
                 are included in this computation are a function of

                   the combined mode algorithm employed and thus not
                   specified in this standard.
                 - The (explicit) ICV field MAY be a part of the ESP
                   packet format when a combined mode algorithm is
                   employed. If one is not used, an analogous field
                   usually will be a part of the ciphertext payload.
                   The location of any integrity fields, and the means
                   by which the Sequence Number and SPI are included in
                   the integrity computation, MUST be defined in an RFC
                   that defines the use of the combined mode algorithm
                   with ESP.

3.3.3. Sequence Number Generation

   The sender's counter is initialized to 0 when an SA is established.
   The sender increments the sequence number (or ESN) counter for this
   SA and inserts the low-order 32 bits of the value into the Sequence
   Number field. Thus, the first packet sent using a given SA will
   contain a sequence number of 1.

   If anti-replay is enabled (the default), the sender checks to ensure
   that the counter has not cycled before inserting the new value in the
   Sequence Number field. In other words, the sender MUST NOT send a
   packet on an SA if doing so would cause the sequence number to cycle.
   An attempt to transmit a packet that would result in sequence number
   overflow is an auditable event. The audit log entry for this event
   SHOULD include the SPI value, current date/time, Source Address,
   Destination Address, and (in IPv6) the cleartext Flow ID.

   The sender assumes anti-replay is enabled as a default, unless
   otherwise notified by the receiver (see Section 3.4.3). Thus,
   typical behavior of an ESP implementation calls for the sender to
   establish a new SA when the Sequence Number (or ESN) cycles, or in
   anticipation of this value cycling.

   If the key used to compute an ICV is manually distributed, a
   compliant implementation SHOULD NOT provide anti-replay service. If
   a user chooses to employ anti-replay in conjunction with SAs that are
   manually keyed, the sequence number counter at the sender MUST be
   correctly maintained across local reboots, etc., until the key is
   replaced. (See Section 5.)

   If anti-replay is disabled (as noted above), the sender does not need
   to monitor or reset the counter. However, the sender still
   increments the counter and when it reaches the maximum value, the
   counter rolls over back to zero. (This behavior is recommended for
   multi-sender, multicast SAs, unless anti-replay mechanisms outside

the scope of this standard are negotiated between the sender and
receiver.)

If ESN (see Appendix) is selected, only the low-order 32 bits of the
sequence number are transmitted in the Sequence Number field,
although both sender and receiver maintain full 64-bit ESN counters.
The high order 32 bits are included in the integrity check in an
algorithm/mode-specific fashion, e.g., the high-order 32 bits may be
appended after the Next Header field when a separate integrity
algorithm is employed.

Note: If a receiver chooses to not enable anti-replay for an SA, then
the receiver SHOULD NOT negotiate ESN in an SA management protocol.
Use of ESN creates a need for the receiver to manage the anti-replay
window (in order to determine the correct value for the high-order
bits of the ESN, which are employed in the ICV computation), which is
generally contrary to the notion of disabling anti-replay for an SA.

3.3.4. Fragmentation

If necessary, fragmentation is performed after ESP processing within
an IPsec implementation. Thus, transport mode ESP is applied only to
whole IP datagrams (not to IP fragments). An IP packet to which ESP
has been applied may itself be fragmented by routers en route, and
such fragments must be reassembled prior to ESP processing at a
receiver. In tunnel mode, ESP is applied to an IP packet, which may
be a fragment of an IP datagram. For example, a security gateway or
a "bump-in-the-stack" or "bump-in-the-wire" IPsec implementation (as
defined in the Security Architecture document) may apply tunnel mode
ESP to such fragments.

NOTE: For transport mode -- As mentioned at the end of Section 3.1.1,
bump-in-the-stack and bump-in-the-wire implementations may have to
first reassemble a packet fragmented by the local IP layer, then
apply IPsec, and then fragment the resulting packet.

NOTE: For IPv6 -- For bump-in-the-stack and bump-in-the-wire
implementations, it will be necessary to examine all the extension
headers to determine if there is a fragmentation header and hence
that the packet needs reassembling prior to IPsec processing.

Fragmentation, whether performed by an IPsec implementation or by
routers along the path between IPsec peers, significantly reduces
performance. Moreover, the requirement for an ESP receiver to accept
fragments for reassembly creates denial of service vulnerabilities.
Thus, an ESP implementation MAY choose to not support fragmentation
and may mark transmitted packets with the DF bit, to facilitate Path
MTU (PMTU) discovery. In any case, an ESP implementation MUST

support generation of ICMP PMTU messages (or equivalent internal
signaling for native host implementations) to minimize the likelihood
of fragmentation. Details of the support required for MTU management
are contained in the Security Architecture document.

3.4. Inbound Packet Processing

3.4.1. Reassembly

If required, reassembly is performed prior to ESP processing. If a
packet offered to ESP for processing appears to be an IP fragment,
i.e., the OFFSET field is non-zero or the MORE FRAGMENTS flag is set,
the receiver MUST discard the packet; this is an auditable event.
The audit log entry for this event SHOULD include the SPI value,
date/time received, Source Address, Destination Address, Sequence
Number, and (in IPv6) the Flow ID.

NOTE: For packet reassembly, the current IPv4 spec does NOT require
either the zeroing of the OFFSET field or the clearing of the MORE
FRAGMENTS flag. In order for a reassembled packet to be processed by
IPsec (as opposed to discarded as an apparent fragment), the IP code
must do these two things after it reassembles a packet.

3.4.2. Security Association Lookup

Upon receipt of a packet containing an ESP Header, the receiver
determines the appropriate (unidirectional) SA via lookup in the SAD.
For a unicast SA, this determination is based on the SPI or the SPI
plus protocol field, as described in Section 2.1. If an
implementation supports multicast traffic, the destination address is
also employed in the lookup (in addition to the SPI), and the sender
address also may be employed, as described in Section 2.1. (This
process is described in more detail in the Security Architecture
document.) The SAD entry for the SA also indicates whether the
Sequence Number field will be checked, whether 32- or 64-bit sequence
numbers are employed for the SA, and whether the (explicit) ICV field
should be present (and if so, its size). Also, the SAD entry will
specify the algorithms and keys to be employed for decryption and ICV
computation (if applicable).

If no valid Security Association exists for this packet, the receiver
MUST discard the packet; this is an auditable event. The audit log
entry for this event SHOULD include the SPI value, date/time
received, Source Address, Destination Address, Sequence Number, and
(in IPv6) the cleartext Flow ID.

(Note that SA management traffic, such as IKE packets, does not need
to be processed based on SPI, i.e., one can demultiplex this traffic
separately based on Next Protocol and Port fields, for example.)

3.4.3. Sequence Number Verification

All ESP implementations MUST support the anti-replay service, though
its use may be enabled or disabled by the receiver on a per-SA basis.
This service MUST NOT be enabled unless the ESP integrity service
also is enabled for the SA, because otherwise the Sequence Number
field has not been integrity protected. Anti-replay is applicable to
unicast as well as multicast SAs. However, this standard specifies
no mechanisms for providing anti-replay for a multi-sender SA
(unicast or multicast). In the absence of negotiation (or manual
configuration) of an anti-replay mechanism for such an SA, it is
recommended that sender and receiver checking of the sequence number
for the SA be disabled (via negotiation or manual configuration), as
noted below.

If the receiver does not enable anti-replay for an SA, no inbound
checks are performed on the Sequence Number. However, from the
perspective of the sender, the default is to assume that anti-replay
is enabled at the receiver. To avoid having the sender do
unnecessary sequence number monitoring and SA setup (see section
3.3.3), if an SA establishment protocol is employed, the receiver
SHOULD notify the sender, during SA establishment, if the receiver
will not provide anti-replay protection.

If the receiver has enabled the anti-replay service for this SA, the
receive packet counter for the SA MUST be initialized to zero when
the SA is established. For each received packet, the receiver MUST
verify that the packet contains a Sequence Number that does not
duplicate the Sequence Number of any other packets received during
the life of this SA. This SHOULD be the first ESP check applied to a
packet after it has been matched to an SA, to speed rejection of
duplicate packets.

ESP permits two-stage verification of packet sequence numbers. This
capability is important whenever an ESP implementation (typically the
cryptographic module portion thereof) is not capable of performing
decryption and/or integrity checking at the same rate as the
interface(s) to unprotected networks. If the implementation is
capable of such "line rate" operation, then it is not necessary to
perform the preliminary verification stage described below.

The preliminary Sequence Number check is effected utilizing the
Sequence Number value in the ESP Header and is performed prior to
integrity checking and decryption. If this preliminary check fails,

the packet is discarded, thus avoiding the need for any cryptographic
operations by the receiver. If the preliminary check is successful,
the receiver cannot yet modify its local counter, because the
integrity of the Sequence Number has not been verified at this point.

Duplicates are rejected through the use of a sliding receive window.
How the window is implemented is a local matter, but the following
text describes the functionality that the implementation must
exhibit.

The "right" edge of the window represents the highest, validated
Sequence Number value received on this SA. Packets that contain
sequence numbers lower than the "left" edge of the window are
rejected. Packets falling within the window are checked against a
list of received packets within the window. If the ESN option is
selected for an SA, only the low-order 32 bits of the sequence number
are explicitly transmitted, but the receiver employs the full
sequence number computed using the high-order 32 bits for the
indicated SA (from his local counter) when checking the received
Sequence Number against the receive window. In constructing the full
sequence number, if the low-order 32 bits carried in the packet are
lower in value than the low-order 32 bits of the receiver's sequence
number, the receiver assumes that the high-order 32 bits have been
incremented, moving to a new sequence number subspace. (This
algorithm accommodates gaps in reception for a single SA as large as
2**32-1 packets. If a larger gap occurs, additional, heuristic
checks for re-synchronization of the receiver sequence number counter
MAY be employed, as described in the Appendix.)

If the received packet falls within the window and is not a
duplicate, or if the packet is to the right of the window, and if a
separate integrity algorithm is employed, then the receiver proceeds
to integrity verification. If a combined mode algorithm is employed,
the integrity check is performed along with decryption. In either
case, if the integrity check fails, the receiver MUST discard the
received IP datagram as invalid; this is an auditable event. The
audit log entry for this event SHOULD include the SPI value,
date/time received, Source Address, Destination Address, the Sequence
Number, and (in IPv6) the Flow ID. The receive window is updated
only if the integrity verification succeeds. (If a combined mode
algorithm is being used, then the integrity protected Sequence Number
must also match the Sequence Number used for anti-replay protection.)

A minimum window size of 32 packets MUST be supported when 32-bit
sequence numbers are employed; a window size of 64 is preferred and
SHOULD be employed as the default. Another window size (larger than
the minimum) MAY be chosen by the receiver. (The receiver does NOT
notify the sender of the window size.) The receive window size

should be increased for higher-speed environments, irrespective of
assurance issues. Values for minimum and recommended receive window
sizes for very high-speed (e.g., multi-gigabit/second) devices are
not specified by this standard.

3.4.4. Integrity Check Value Verification

   As with outbound processing, there are several options for inbound
   processing, based on features of the algorithms employed.

3.4.4.1. Separate Confidentiality and Integrity Algorithms

   If separate confidentiality and integrity algorithms are employed
   processing proceeds as follows:

        1. If integrity has been selected, the receiver computes the
           ICV over the ESP packet minus the ICV, using the specified
           integrity algorithm and verifies that it is the same as the
           ICV carried in the packet. Details of the computation are
           provided below.

           If the computed and received ICVs match, then the datagram
           is valid, and it is accepted. If the test fails, then the
           receiver MUST discard the received IP datagram as invalid;
           this is an auditable event. The log data SHOULD include the
           SPI value, date/time received, Source Address, Destination
           Address, the Sequence Number, and (for IPv6) the cleartext
           Flow ID.

           Implementation Note:

           Implementations can use any set of steps that results in the
           same result as the following set of steps. Begin by
           removing and saving the ICV field. Next check the overall
           length of the ESP packet minus the ICV field. If implicit
           padding is required, based on the block size of the
           integrity algorithm, append zero-filled bytes to the end of
           the ESP packet directly after the Next Header field, or
           after the high-order 32 bits of the sequence number if ESN
           is selected. Perform the ICV computation and compare the
           result with the saved value, using the comparison rules
           defined by the algorithm specification.

        2. The receiver decrypts the ESP Payload Data, Padding, Pad
           Length, and Next Header using the key, encryption algorithm,
           algorithm mode, and cryptographic synchronization data (if
           any), indicated by the SA. As in Section 3.3.2, we speak
           here in terms of encryption always being applied because of

the formatting implications. This is done with the
understanding that "no confidentiality" is offered by using
the NULL encryption algorithm (RFC 2410).

- If explicit cryptographic synchronization data, e.g.,
  an IV, is indicated, it is taken from the Payload
  field and input to the decryption algorithm as per
  the algorithm specification.

- If implicit cryptographic synchronization data is
  indicated, a local version of the IV is constructed
  and input to the decryption algorithm as per the
  algorithm specification.

3. The receiver processes any Padding as specified in the
   encryption algorithm specification. If the default padding
   scheme (see Section 2.4) has been employed, the receiver
   SHOULD inspect the Padding field before removing the padding
   prior to passing the decrypted data to the next layer.

4. The receiver checks the Next Header field. If the value is
   "59" (no next header), the (dummy) packet is discarded
   without further processing.

5. The receiver reconstructs the original IP datagram from:

   - for transport mode -- outer IP header plus the
     original next layer protocol information in the ESP
     Payload field
   - for tunnel mode -- the entire IP datagram in the ESP
     Payload field.

   The exact steps for reconstructing the original datagram
   depend on the mode (transport or tunnel) and are described
   in the Security Architecture document. At a minimum, in an
   IPv6 context, the receiver SHOULD ensure that the decrypted
   data is 8-byte aligned, to facilitate processing by the
   protocol identified in the Next Header field. This
   processing "discards" any (optional) TFC padding that has
   been added for traffic flow confidentiality. (If present,
   this will have been inserted after the IP datagram (or
   transport-layer frame) and before the Padding field (see
   Section 2.4).)

If integrity checking and encryption are performed in parallel,
integrity checking MUST be completed before the decrypted packet is
passed on for further processing. This order of processing
facilitates rapid detection and rejection of replayed or bogus

packets by the receiver, prior to decrypting the packet, hence potentially reducing the impact of denial of service attacks.

Note: If the receiver performs decryption in parallel with integrity checking, care must be taken to avoid possible race conditions with regard to packet access and extraction of the decrypted packet.

3.4.4.2. Combined Confidentiality and Integrity Algorithms

If a combined confidentiality and integrity algorithm is employed, then the receiver proceeds as follows:

1. Decrypts and integrity checks the ESP Payload Data, Padding, Pad Length, and Next Header, using the key, algorithm, algorithm mode, and cryptographic synchronization data (if any), indicated by the SA. The SPI from the ESP header, and the (receiver) packet counter value (adjusted as required from the processing described in Section 3.4.3) are inputs to this algorithm, as they are required for the integrity check.

    - If explicit cryptographic synchronization data, e.g., an IV, is indicated, it is taken from the Payload field and input to the decryption algorithm as per the algorithm specification.

    - If implicit cryptographic synchronization data, e.g., an IV, is indicated, a local version of the IV is constructed and input to the decryption algorithm as per the algorithm specification.

2. If the integrity check performed by the combined mode algorithm fails, the receiver MUST discard the received IP datagram as invalid; this is an auditable event. The log data SHOULD include the SPI value, date/time received, Source Address, Destination Address, the Sequence Number, and (in IPv6) the cleartext Flow ID.

3. Process any Padding as specified in the encryption algorithm specification, if the algorithm has not already done so.

4. The receiver checks the Next Header field. If the value is "59" (no next header), the (dummy) packet is discarded without further processing.

5. Extract the original IP datagram (tunnel mode) or transport-layer frame (transport mode) from the ESP Payload Data field. This implicitly discards any (optional) padding

that has been added for traffic flow confidentiality. (If
present, the TFC padding will have been inserted after the
IP payload and before the Padding field (see Section 2.4).)

4. Auditing

   Not all systems that implement ESP will implement auditing. However,
   if ESP is incorporated into a system that supports auditing, then the
   ESP implementation MUST also support auditing and MUST allow a system
   administrator to enable or disable auditing for ESP. For the most
   part, the granularity of auditing is a local matter. However,
   several auditable events are identified in this specification and for
   each of these events a minimum set of information that SHOULD be
   included in an audit log is defined.

        - No valid Security Association exists for a session. The
          audit log entry for this event SHOULD include the SPI value,
          date/time received, Source Address, Destination Address,
          Sequence Number, and (for IPv6) the cleartext Flow ID.

        - A packet offered to ESP for processing appears to be an IP
          fragment, i.e., the OFFSET field is non-zero or the MORE
          FRAGMENTS flag is set. The audit log entry for this event
          SHOULD include the SPI value, date/time received, Source
          Address, Destination Address, Sequence Number, and (in IPv6)
          the Flow ID.

        - Attempt to transmit a packet that would result in Sequence
          Number overflow. The audit log entry for this event SHOULD
          include the SPI value, current date/time, Source Address,
          Destination Address, Sequence Number, and (for IPv6) the
          cleartext Flow ID.

        - The received packet fails the anti-replay checks. The audit
          log entry for this event SHOULD include the SPI value,
          date/time received, Source Address, Destination Address, the
          Sequence Number, and (in IPv6) the Flow ID.

        - The integrity check fails. The audit log entry for this
          event SHOULD include the SPI value, date/time received,
          Source Address, Destination Address, the Sequence Number, and
          (for IPv6) the Flow ID.

   Additional information also MAY be included in the audit log for each
   of these events, and additional events, not explicitly called out in
   this specification, also MAY result in audit log entries. There is
   no requirement for the receiver to transmit any message to the

purported sender in response to the detection of an auditable event,
because of the potential to induce denial of service via such action.

5. Conformance Requirements

   Implementations that claim conformance or compliance with this
   specification MUST implement the ESP syntax and processing described
   here for unicast traffic, and MUST comply with all additional packet
   processing requirements levied by the Security Architecture document
   [Ken-Arch]. Additionally, if an implementation claims to support
   multicast traffic, it MUST comply with the additional requirements
   specified for support of such traffic. If the key used to compute an
   ICV is manually distributed, correct provision of the anti-replay
   service requires correct maintenance of the counter state at the
   sender (across local reboots, etc.), until the key is replaced, and
   there likely would be no automated recovery provision if counter
   overflow were imminent. Thus, a compliant implementation SHOULD NOT
   provide anti-replay service in conjunction with SAs that are manually
   keyed.

   The mandatory-to-implement algorithms for use with ESP are described
   in a separate document [Eas04], to facilitate updating the algorithm
   requirements independently from the protocol per se. Additional
   algorithms, beyond those mandated for ESP, MAY be supported.

   Because use of encryption in ESP is optional, support for the "NULL"
   encryption algorithm also is required to maintain consistency with
   the way ESP services are negotiated. Support for the
   confidentiality-only service version of ESP is optional. If an
   implementation offers this service, it MUST also support the
   negotiation of the "NULL" integrity algorithm. NOTE that although
   integrity and encryption may each be "NULL" under the circumstances
   noted above, they MUST NOT both be "NULL".

6. Security Considerations

   Security is central to the design of this protocol, and thus security
   considerations permeate the specification. Additional security-
   relevant aspects of using the IPsec protocol are discussed in the
   Security Architecture document.

7. Differences from RFC 2406

   This document differs from RFC 2406 in a number of significant ways.

        o Confidentiality-only service -- now a MAY, not a MUST.

        o SPI -- modified to specify a uniform algorithm for SAD lookup
          for unicast and multicast SAs, covering a wider range of
          multicast technologies. For unicast, the SPI may be used
          alone to select an SA, or may be combined with the protocol,
          at the option of the receiver. For multicast SAs, the SPI is
          combined with the destination address, and optionally the
          source address, to select an SA.
        o Extended Sequence Number -- added a new option for a 64-bit
          sequence number for very high-speed communications. Clarified
          sender and receiver processing requirements for multicast SAs
          and multi-sender SAs.
        o Payload data -- broadened model to accommodate combined mode
          algorithms.
        o Padding for improved traffic flow confidentiality -- added
          requirement to be able to add bytes after the end of the IP
          Payload, prior to the beginning of the Padding field.
        o Next Header -- added requirement to be able to generate and
          discard dummy padding packets (Next Header = 59)
        o ICV -- broadened model to accommodate combined mode
          algorithms.
        o Algorithms -- Added combined confidentiality mode algorithms.
        o Moved references to mandatory algorithms to a separate
          document.
        o Inbound and Outbound packet processing -- there are now two
          paths: (1) separate confidentiality and integrity
          algorithms and (2) combined confidentiality mode
          algorithms. Because of the addition of combined mode
          algorithms, the encryption/decryption and integrity sections
          have been combined for both inbound and outbound packet
          processing.

8. Backward-Compatibility Considerations

   There is no version number in ESP and no mechanism enabling IPsec
   peers to discover or negotiate which version of ESP each is using or
   should use. This section discusses consequent backward-compatibility
   issues.

   First, if none of the new features available in ESP v3 are employed,
   then the format of an ESP packet is identical in ESP v2 and v3. If a
   combined mode encryption algorithm is employed, a feature supported
   only in ESP v3, then the resulting packet format may differ from the
   ESP v2 spec. However, a peer who implements only ESP v2 would never
   negotiate such an algorithm, as they are defined for use only in the
   ESP v3 context.

Extended Sequence Number (ESN) negotiation is supported by IKE v2 and
has been addressed for IKE v1 by the ESN Addendum to the IKE v1
Domain of Interpretation (DOI).

In the new ESP (v3), we make two provisions to better support traffic
flow confidentiality (TFC):

    - arbitrary padding after the end of an IP packet
    - a discard convention using Next Header = 59

The first feature is one that should not cause problems for a
receiver, since the IP total length field indicates where the IP
packet ends. Thus, any TFC padding bytes after the end of the packet
should be removed at some point during IP packet processing, after
ESP processing, even if the IPsec software does not remove such
padding. Thus, this is an ESP v3 feature that a sender can employ
irrespective of whether a receiver implements ESP v2 or ESP v3.

The second feature allows a sender to send a payload that is an
arbitrary string of bytes that do not necessarily constitute a well-
formed IP packet, inside of a tunnel, for TFC purposes. It is an
open question as to what an ESP v2 receiver will do when the Next
Header field in an ESP packet contains the value "59". It might
discard the packet when it finds an ill-formed IP header, and log
this event, but it certainly ought not to crash, because such
behavior would constitute a DoS vulnerability relative to traffic
received from authenticated peers. Thus this feature is an
optimization that an ESP v3 sender can make use of irrespective of
whether a receiver implements ESP v2 or ESP v3.

9. Acknowledgements

The author would like to acknowledge the contributions of Ran
Atkinson, who played a critical role in initial IPsec activities, and
who authored the first series of IPsec standards: RFCs 1825-1827.
Karen Seo deserves special thanks for providing help in the editing
of this and the previous version of this specification. The author
also would like to thank the members of the IPSEC and MSEC working
groups who have contributed to the development of this protocol
specification.

10. References

10.1. Normative References

   [Bra97]   Bradner, S., "Key words for use in RFCs to Indicate
               Requirement Level", BCP 14, RFC 2119, March 1997.

   [DH98]   Deering, S. and R. Hinden, "Internet Protocol, Version 6
                  (IPv6) Specification", RFC 2460, December 1998.

   [Eas04]   3rd Eastlake, D., "Cryptographic Algorithm Implementation
                  Requirements for Encapsulating Security Payload (ESP) and
                  Authentication Header (AH)", RFC 4305, December 2005.

   [Ken-Arch]   Kent, S. and K. Seo, "Security Architecture for the
                  Internet Protocol", RFC 4301, December 2005.

   [Pos81]   Postel, J., "Internet Protocol", STD 5, RFC 791, September
                  1981.

10.2. Informative References

   [Bel96]   Steven M. Bellovin, "Problem Areas for the IP Security
                  Protocols", Proceedings of the Sixth Usenix Unix Security
                  Symposium, July, 1996.

   [HC03]   Holbrook, H. and B. Cain, "Source-Specific Multicast for
                  IP", Work in Progress, November 3, 2002.

   [Kau05]   Kaufman, C., Ed., "The Internet Key Exchange (IKEv2)
                  Protocol", RFC 4306, December 2005.

   [Ken-AH]   Kent, S., "IP Authentication Header", RFC 4302, December
                  2005.

   [Kra01]   Krawczyk, H., "The Order of Encryption and Authentication
                  for Protecting Communications (Or: How Secure Is SSL?)",
                  CRYPTO' 2001.

   [NIST01]   Federal Information Processing Standards Publication 140-2
                  (FIPS PUB 140-2), "Security Requirements for Cryptographic
                  Modules", Information Technology Laboratory, National
                  Institute of Standards and Technology, May 25, 2001.

   [RFC3547]   Baugher, M., Weis, B., Hardjono, T., and H. Harney, "The
                  Group Domain of Interpretation", RFC 3547, July 2003.

   [RFC3740]   Hardjono, T. and B. Weis, "The Multicast Group Security
                  Architecture", RFC 3740, March 2004.

   [Syverson]   P. Syverson, D. Goldschlag, and M. Reed, "Anonymous
                  Connections and Onion Routing", Proceedings of the
                  Symposium on Security and Privacy, Oakland, CA, May 1997,
                  pages 44-54.

Appendix A: Extended (64-bit) Sequence Numbers

A1. Overview

   This appendix describes an extended sequence number (ESN) scheme for
   use with IPsec (ESP and AH) that employs a 64-bit sequence number,
   but in which only the low-order 32 bits are transmitted as part of
   each packet. It covers both the window scheme used to detect
   replayed packets and the determination of the high-order bits of the
   sequence number that are used both for replay rejection and for
   computation of the ICV. It also discusses a mechanism for handling
   loss of synchronization relative to the (not transmitted) high-order
   bits.

A2. Anti-Replay Window

   The receiver will maintain an anti-replay window of size W. This
   window will limit how far out of order a packet can be, relative to
   the packet with the highest sequence number that has been
   authenticated so far. (No requirement is established for minimum or
   recommended sizes for this window, beyond the 32- and 64-packet
   values already established for 32-bit sequence number windows.
   However, it is suggested that an implementer scale these values
   consistent with the interface speed supported by an implementation
   that makes use of the ESN option. Also, the algorithm described
   below assumes that the window is no greater than 2^{31} packets in
   width.) All 2^{32} sequence numbers associated with any fixed value
   for the high-order 32 bits (Seqh) will hereafter be called a sequence
   number subspace. The following table lists pertinent variables and
   their definitions.

Var. Name	Size (bits)	Meaning
W	32	Size of window
T	64	Highest sequence number authenticated so far, upper bound of window
Tl	32	Lower 32 bits of T
Th	32	Upper 32 bits of T
B	64	Lower bound of window
Bl	32	Lower 32 bits of B
Bh	32	Upper 32 bits of B
Seq	64	Sequence Number of received packet
Seql	32	Lower 32 bits of Seq
Seqh	32	Upper 32 bits of Seq

When performing the anti-replay check, or when determining which
high-order bits to use to authenticate an incoming packet, there are
two cases:

   + Case A: Tl >= (W - 1). In this case, the window is within one
                            sequence number subspace. (See Figure 1)
   + Case B: Tl < (W - 1). In this case, the window spans two
                            sequence number subspaces. (See Figure 2)

In the figures below, the bottom line ("----") shows two consecutive
sequence number subspaces, with zeros indicating the beginning of
each subspace. The two shorter lines above it show the higher-order
bits that apply. The "====" represents the window. The "****"
represents future sequence numbers, i.e., those beyond the current
highest sequence number authenticated (ThTl).
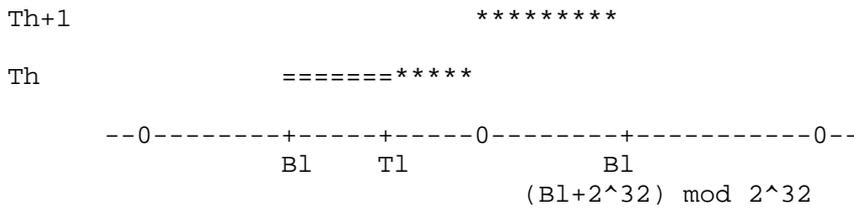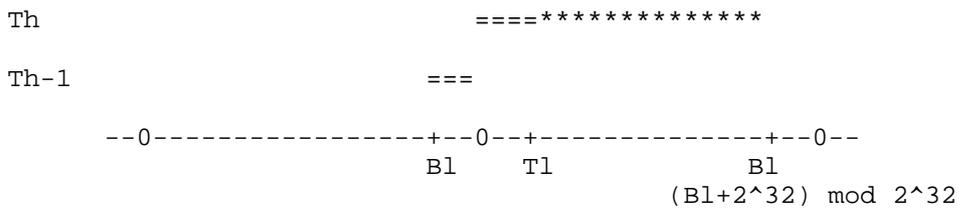
```
        Th+1                            *********

        Th                    =======*****

            --0--------+-----+-----0--------+-----------0--
                      Bl    Tl          Bl
                                    (Bl+2^32) mod 2^32

                       Figure 1 -- Case A


        Th                           ====**************

        Th-1                         ===

            --0----------------+--0--+-------------+--0--
                      Bl    Tl          Bl
                                    (Bl+2^32) mod 2^32

                       Figure 2 -- Case B
```

A2.1. Managing and Using the Anti-Replay Window

   The anti-replay window can be thought of as a string of bits where
   'W' defines the length of the string. W = T - B + 1 and cannot
   exceed 2^32 - 1 in value. The bottom-most bit corresponds to B and
   the top-most bit corresponds to T, and each sequence number from Bl
   through Tl is represented by a corresponding bit. The value of the
   bit indicates whether or not a packet with that sequence number has
   been received and authenticated, so that replays can be detected and
   rejected.

When a packet with a 64-bit sequence number (Seq) greater than T is
received and validated,

    + B is increased by (Seq - T)
    + (Seq - T) bits are dropped from the low end of the window
    + (Seq - T) bits are added to the high end of the window
    + The top bit is set to indicate that a packet with that sequence
      number has been received and authenticated
    + The new bits between T and the top bit are set to indicate that
      no packets with those sequence numbers have been received yet.
    + T is set to the new sequence number

In checking for replayed packets,

    + Under Case A: If Seql >= Bl (where Bl = Tl - W + 1) AND Seql <=
      Tl, then check the corresponding bit in the window to see if
      this Seql has already been seen. If yes, reject the packet. If
      no, perform integrity check (see Appendix A2.2. below for
      determination of Seqh).

    + Under Case B: If Seql >= Bl (where Bl = Tl - W + 1) OR Seql <=
      Tl, then check the corresponding bit in the window to see if
      this Seql has already been seen. If yes, reject the packet. If
      no, perform integrity check (see Appendix A2.2. below for
      determination of Seqh).

A2.2. Determining the Higher-Order Bits (Seqh) of the Sequence Number

Because only 'Seql' will be transmitted with the packet, the receiver
must deduce and track the sequence number subspace into which each
packet falls, i.e., determine the value of Seqh. The following
equations define how to select Seqh under "normal" conditions; see
Section A3 for a discussion of how to recover from extreme packet
loss.

    + Under Case A (Figure 1):
      If Seql >= Bl (where Bl = Tl - W + 1), then Seqh = Th
      If Seql < Bl (where Bl = Tl - W + 1), then Seqh = Th + 1

    + Under Case B (Figure 2):
      If Seql >= Bl (where Bl = Tl - W + 1), then Seqh = Th - 1
      If Seql < Bl (where Bl = Tl - W + 1), then Seqh = Th

A2.3. Pseudo-Code Example

   The following pseudo-code illustrates the above algorithms for anti-
   replay and integrity checks. The values for 'Seql', 'Tl', 'Th' and
   'W' are 32-bit unsigned integers. Arithmetic is mod 2^32.

```
        If (Tl >= W - 1)                          Case A
            If (Seql >= Tl - W + 1)
                Seqh = Th
                If (Seql <= Tl)
                    If (pass replay check)
                        If (pass integrity check)
                            Set bit corresponding to Seql
                            Pass the packet on
                        Else reject packet
                    Else reject packet
                Else
                    If (pass integrity check)
                        Tl = Seql (shift bits)
                        Set bit corresponding to Seql
                        Pass the packet on
                    Else reject packet
            Else
                Seqh = Th + 1
                If (pass integrity check)
                    Tl = Seql (shift bits)
                    Th = Th + 1
                    Set bit corresponding to Seql
                    Pass the packet on
                Else reject packet
        Else                                      Case B
            If (Seql >= Tl - W + 1)
                Seqh = Th - 1
                If (pass replay check)
                    If (pass integrity check)
                        Set the bit corresponding to Seql
                        Pass packet on
                    Else reject packet
                Else reject packet
            Else
                Seqh = Th
                If (Seql <= Tl)
                    If (pass replay check)
                        If (pass integrity check)
                            Set the bit corresponding to Seql
                            Pass packet on
                        Else reject packet
                    Else reject packet
```

```
              Else
                   If (pass integrity check)
                        Tl = Seql (shift bits)
                        Set the bit corresponding to Seql
                        Pass packet on
                   Else reject packet
```

A3. Handling Loss of Synchronization due to Significant Packet Loss

   If there is an undetected packet loss of 2^32 or more consecutive
   packets on a single SA, then the transmitter and receiver will lose
   synchronization of the high-order bits, i.e., the equations in
   Section A2.2. will fail to yield the correct value. Unless this
   problem is detected and addressed, subsequent packets on this SA will
   fail authentication checks and be discarded. The following procedure
   SHOULD be implemented by any IPsec (ESP or AH) implementation that
   supports the ESN option.

   Note that this sort of extended traffic loss is likely to be detected
   at higher layers in most cases, before IPsec would have to invoke the
   sort of re-synchronization mechanism described in A3.1 and A3.2. If
   any significant fraction of the traffic on the SA in question is TCP,
   the source would fail to receive ACKs and would stop sending long
   before 2^32 packets had been lost. Also, for any bi-directional
   application, even ones operating above UDP, such an extended outage
   would likely result in triggering some form of timeout. However, a
   unidirectional application, operating over UDP, might lack feedback
   that would cause automatic detection of a loss of this magnitude,
   hence the motivation to develop a recovery method for this case.
   Note that the above observations apply to SAs between security
   gateways, or between hosts, or between host and security gateways.

   The solution we've chosen was selected to:

     + minimize the impact on normal traffic processing

     + avoid creating an opportunity for a new denial of service attack
       such as might occur by allowing an attacker to force diversion of
       resources to a re-synchronization process

     + limit the recovery mechanism to the receiver -- because anti-
       replay is a service only for the receiver, and the transmitter
       generally is not aware of whether the receiver is using sequence
       numbers in support of this optional service, it is preferable for
       recovery mechanisms to be local to the receiver. This also
       allows for backward compatibility.

A3.1. Triggering Re-synchronization

   For each SA, the receiver records the number of consecutive packets
   that fail authentication. This count is used to trigger the re-
   synchronization process, which should be performed in the background
   or using a separate processor. Receipt of a valid packet on the SA
   resets the counter to zero. The value used to trigger the re-
   synchronization process is a local parameter. There is no
   requirement to support distinct trigger values for different SAs,
   although an implementer may choose to do so.

A3.2. Re-synchronization Process

   When the above trigger point is reached, a "bad" packet is selected
   for which authentication is retried using successively larger values
   for the upper half of the sequence number (Seqh). These values are
   generated by incrementing by one for each retry. The number of
   retries should be limited, in case this is a packet from the "past"
   or a bogus packet. The limit value is a local parameter. (Because
   the Seqh value is implicitly placed after the ESP (or AH) payload, it
   may be possible to optimize this procedure by executing the integrity
   algorithm over the packet up to the endpoint of the payload, then
   compute different candidate ICVs by varying the value of Seqh.)
   Successful authentication of a packet via this procedure resets the
   consecutive failure count and sets the value of T to that of the
   received packet.

   This solution requires support only on the part of the receiver,
   thereby allowing for backward compatibility. Also, because re-
   synchronization efforts would either occur in the background or
   utilize an additional processor, this solution does not impact
   traffic processing and a denial of service attack cannot divert
   resources away from traffic processing.

Author's Address

   Stephen Kent
   BBN Technologies
   10 Moulton Street
   Cambridge, MA 02138
   USA

   Phone: +1 (617) 873-3988
   EMail: kent@bbn.com

Full Copyright Statement

Intellectual Property

Acknowledgement