                    The ESP CBC-Mode Cipher Algorithms

Status of this Memo

Copyright Notice

Abstract

   This document describes how to use CBC-mode cipher algorithms with
   the IPSec ESP (Encapsulating Security Payload) Protocol. It not only
   clearly states how to use certain cipher algorithms, but also how to
   use all CBC-mode cipher algorithms.

Table of Contents

1. Introduction

   The Encapsulating Security Payload (ESP) [Kent98] provides
   confidentiality for IP datagrams by encrypting the payload data to be
   protected. This specification describes the ESP use of CBC-mode
   cipher algorithms.

   While this document does not describe the use of the default cipher
   algorithm DES, the reader should be familiar with that document.
   [Madson98]

   It is assumed that the reader is familiar with the terms and concepts
   described in the "Security Architecture for the Internet Protocol"
   [Atkinson95], "IP Security Document Roadmap" [Thayer97], and "IP
   Encapsulating Security Payload (ESP)" [Kent98] documents.

   Furthermore, this document is a companion to [Kent98] and MUST be
   read in its context.

1.1 Specification of Requirements

   The keywords "MUST", "MUST NOT", "REQUIRED", "SHOULD", "SHOULD NOT",
   and "MAY" that appear in this document are to be interpreted as
   described in [Bradner97].

1.2 Intellectual Property Rights Statement

   The IETF takes no position regarding the validity or scope of any
   intellectual property or other rights that might be claimed to
   pertain to the implementation or use of the technology described in
   this document or the extent to which any license under such rights
   might or might not be available; neither does it represent that it
   has made any effort to identify any such rights. Information on the
   IETF's procedures with respect to rights in standards-track and
   standards-related documentation can be found in BCP-11. Copies of
   claims of rights made available for publication and any assurances of
   licenses to be made available, or the result of an attempt made to
   obtain a general license or permission for the use of such
   proprietary rights by implementers or users of this specification can
   be obtained from the IETF Secretariat.

2. Cipher Algorithms

   All symmetric block cipher algorithms share common characteristics
   and variables. These include mode, key size, weak keys, block size,
   and rounds. All of which will be explained below.

   While this document illustrates certain cipher algorithms such as
   Blowfish [Schneier93], CAST-128 [Adams97], 3DES, IDEA [Lai] [MOV],
   and RC5 [Baldwin96], any other block cipher algorithm may be used
   with ESP if all of the variables described within this document are
   clearly defined.

2.1 Mode

   All symmetric block cipher algorithms described or insinuated within
   this document use Cipher Block Chaining (CBC) mode. This mode
   requires an Initialization Vector (IV) that is the same size as the
   block size. Use of a randomly generated IV prevents generation of
   identical ciphertext from packets which have identical data that
   spans the first block of the cipher algorithm's blocksize.

   The IV is XOR'd with the first plaintext block, before it is
   encrypted. Then for successive blocks, the previous ciphertext block
   is XOR'd with the current plaintext, before it is encrypted.

   More information on CBC mode can be obtained in [Schneier95].

2.2 Key Size

   Some cipher algorithms allow for variable sized keys, while others
   only allow a specific key size. The length of the key correlates
   with the strength of that algorithm, thus larger keys are always
   harder to break than shorter ones.

   This document stipulates that all key sizes MUST be a multiple of 8
   bits.

   This document does specify the default key size for each cipher
   algorithm. This size was chosen by consulting experts on the
   algorithm and by balancing strength of the algorithm with
   performance.

```
+==============+=================+================+==========+
| Algorithm    | Key Sizes (bits) | Popular Sizes  | Default  |
+==============+=================+================+==========+
| CAST-128 [1] | 40 to 128       | 40, 64, 80, 128 | 128     |
+--------------+-----------------+----------------+----------+
| RC5          | 40 to 2040      | 40, 128, 160    | 128     |
+--------------+-----------------+----------------+----------+
| IDEA         | 128             | 128             | 128     |
+--------------+-----------------+----------------+----------+
| Blowfish     | 40 to 448       | 128             | 128     |
+--------------+-----------------+----------------+----------+
| 3DES [2]     | 192             | 192             | 192     |
+--------------+-----------------+----------------+----------+
```

   Notes:

   [1] With CAST-128, keys less than 128 bits MUST be padded with zeros
   in the rightmost, or least significant, positions out to 128 bits
   since the CAST-128 key schedule assumes an input key of 128 bits.
   Thus if you had a key with a size of 80 bits '3B5D831CFE', it would
   be padded to produce a key with a size of 128 bits
   '3B5D831CFE000000'.

   [2] The first 3DES key is taken from the first 64 bits, the second
   from the next 64 bits, and the third from the last 64 bits.
   Implementations MUST take into consideration the parity bits when
   initially accepting a new set of keys. Each of the three keys is
   really 56 bits in length with the extra 8 bits used for parity.

   The reader should note that the minimum key size for all of the above
   cipher algorithms is 40 bits, and that the authors strongly advise
   that implementations do NOT use key sizes smaller than 40 bits.

2.3 Weak Keys

   Weak key checks SHOULD be performed. If such a key is found, the key
   SHOULD be rejected and a new SA requested. Some cipher algorithms
   have weak keys or keys that MUST not be used due to their weak
   nature.

   New weak keys might be discovered, so this document does not in any
   way contain all possible weak keys for these ciphers. Please check
   with other sources of cryptography such as [MOV] and [Schneier] for
   further weak keys.

   CAST-128:

   No known weak keys.

   RC5:

   No known weak keys when used with 16 rounds.

    IDEA:

   IDEA has been found to have weak keys. Please check with [MOV] and
   [Schneier] for more information.

   Blowfish:

   Weak keys for Blowfish have been discovered. Weak keys are keys that
   produce the identical entries in a given S-box. Unfortunately, there
   is no way to test for weak keys before the S- box values are
   generated. However, the chances of randomly generating such a key
   are small.

   3DES:

   DES has 64 known weak keys, including so-called semi-weak keys and
   possibly-weak keys [Schneier95, pp 280-282]. The likelihood of
   picking one at random is negligible.

   For DES-EDE3, there is no known need to reject weak or
   complementation keys. Any weakness is obviated by the use of
   multiple keys.

   However, if the first two or last two independent 64-bit keys are
   equal (k1 == k2 or k2 == k3), then the 3DES operation is simply the
   same as DES. Implementers MUST reject keys that exhibit this
   property.

2.4 Block Size and Padding

   All of the algorithms described in this document use a block size of
   eight octets (64 bits).

   Padding is used to align the payload type and pad length octets as
   specified in [Kent98]. Padding must be sufficient to align the data
   to be encrypted to an eight octet (64 bit) boundary.

2.5 Rounds

   This variable determines how many times a block is encrypted. While
   this variable MAY be negotiated, a default value MUST always exist
   when it is not negotiated.

   +=====================+============+=====================+
   | Algorithm | Negotiable | Default Rounds |
   +=====================+============+=====================+
   | CAST-128 | No | key<=80 bits, 12 |
   | | | key>80 bits, 16 |
   +---------------------+------------+---------------------+
   | RC5 | No | 16 |
   +---------------------+------------+---------------------+
   | IDEA | No | 8 |
   +---------------------+------------+---------------------+
   | Blowfish | No | 16 |
   +---------------------+------------+---------------------+
   | 3DES | No | 48 (16x3) |
   +---------------------+------------+---------------------+

2.6 Backgrounds

   CAST-128:

   The CAST design procedure was originally developed by Carlisle Adams
   and Stafford Tavares at Queen's University, Kingston, Ontario,
   Canada. Subsequent enhancements have been made over the years by
   Carlisle Adams and Michael Wiener of Entrust Technologies. CAST-128
   is the result of applying the CAST Design Procedure as outlined in
   [Adams97].

   RC5:

   The RC5 encryption algorithm was developed by Ron Rivest for RSA Data
   Security Inc. in order to address the need for a high- performance
   software and hardware ciphering alternative to DES. It is patented
   (pat.no. 5,724,428). A description of RC5 may be found in [MOV] and
   [Schneier].

   IDEA:

   Xuejia Lai and James Massey developed the IDEA (International Data
   Encryption Algorithm) algorithm. The algorithm is described in
   detail in [Lai], [Schneier] and [MOV].

The IDEA algorithm is patented in Europe and in the United States
with patent application pending in Japan. Licenses are required for
commercial uses of IDEA.

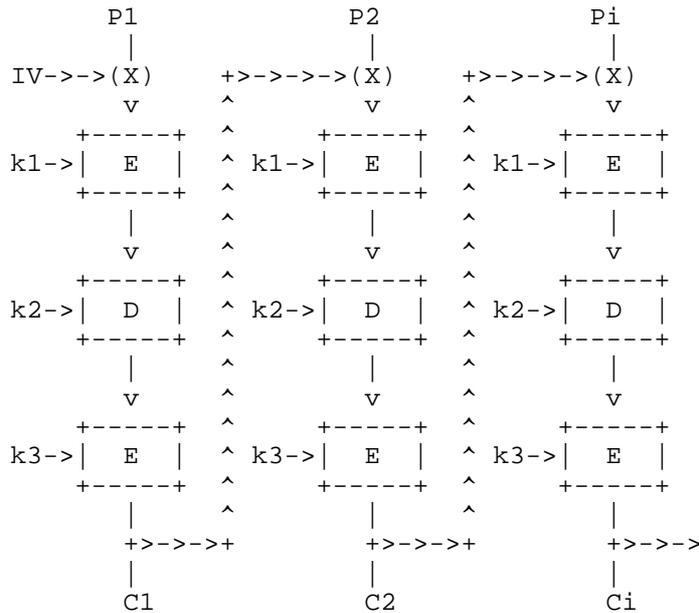For patent and licensing information, contact:

        Ascom Systec AG, Dept. CMVV
        Gewerbepark, CH-5506
        Magenwil, Switzerland
        Phone: +41 64 56 59 83
        Fax: +41 64 56 59 90
        idea@ascom.ch
        http://www.ascom.ch/Web/systec/policy/normal/exhibit1.html

Blowfish:

Bruce Schneier of Counterpane Systems developed the Blowfish block
cipher algorithm. The algorithm is described in detail in
[Schneier93], [Schneier95] and [Schneier].

3DES:

This DES variant, colloquially known as "Triple DES" or as DES-EDE3,
processes each block three times, each time with a different key.
This technique of using more than one DES operation was proposed in
[Tuchman79].

```
                  P1              P2              Pi
                  |               |               |
              IV->->(X)    +>->->->(X)    +>->->->(X)
                   v       ^       v       ^       v
                +-----+    ^    +-----+    ^    +-----+
            k1->|  E  |    ^ k1->|  E  |   ^ k1->|  E  |
                +-----+    ^    +-----+    ^    +-----+
                   |       ^       |       ^       |
                   v       ^       v       ^       v
                +-----+    ^    +-----+    ^    +-----+
            k2->|  D  |    ^ k2->|  D  |   ^ k2->|  D  |
                +-----+    ^    +-----+    ^    +-----+
                   |       ^       |       ^       |
                   v       ^       v       ^       v
                +-----+    ^    +-----+    ^    +-----+
            k3->|  E  |    ^ k3->|  E  |   ^ k3->|  E  |
                +-----+    ^    +-----+    ^    +-----+
                   |       ^       |       ^       |
                  +>->->+         +>->->+         +>->->
                   |               |               |
                   C1              C2              Ci
```

The DES-EDE3-CBC algorithm is a simple variant of the DES-CBC
algorithm [FIPS-46]. The "outer" chaining technique is used.

In DES-EDE3-CBC, an Initialization Vector (IV) is XOR'd with the
first 64-bit (8 byte) plaintext block (P1). The keyed DES function
is iterated three times, an encryption (Ek1) followed by a decryption
(Dk2) followed by an encryption (Ek3), and generates the ciphertext
(C1) for the block. Each iteration uses an independent key: k1, k2
and k3.

For successive blocks, the previous ciphertext block is XOR'd with
the current plaintext (Pi). The keyed DES-EDE3 encryption function
generates the ciphertext (Ci) for that block.

To decrypt, the order of the functions is reversed: decrypt with k3,
encrypt with k2, decrypt with k1, and XOR the previous ciphertext
block.

Note that when all three keys (k1, k2 and k3) are the same, DES-
EDE3-CBC is equivalent to DES-CBC. This property allows the DES-EDE3
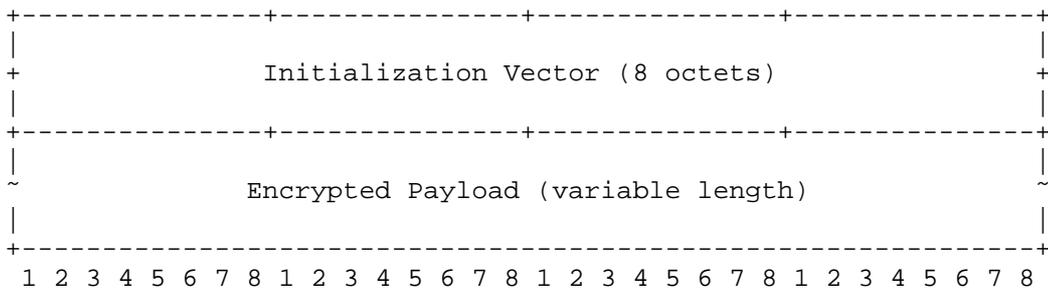hardware implementations to operate in DES mode without modification.

For more explanation and implementation information for Triple DES,
see [Schneier95].

2.7 Performance

For a comparison table of the estimated speed of any of these and
other cipher algorithms, please see [Schneier97] or for an up-to-date
performance comparison, please see [Bosseleaers].

3. ESP Payload

The ESP payload is made up of the IV followed by raw cipher-text.
Thus the payload field, as defined in [Kent98], is broken down
according to the following diagram:

```
+---------------+---------------+---------------+---------------+
|                                                               |
+             Initialization Vector (8 octets)                 +
|                                                               |
+---------------+---------------+---------------+---------------+
|                                                               |
~             Encrypted Payload (variable length)              ~
|                                                               |
+---------------------------------------------------------------+
 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8 1 2 3 4 5 6 7 8
```

The IV field MUST be same size as the block size of the cipher
algorithm being used. The IV MUST be chosen at random. Common
practice is to use random data for the first IV and the last block of
encrypted data from an encryption process as the IV for the next
encryption process.

Including the IV in each datagram ensures that decryption of each
received datagram can be performed, even when some datagrams are
dropped, or datagrams are re-ordered in transit.

To avoid ECB encryption of very similar plaintext blocks in different
packets, implementations MUST NOT use a counter or other low-Hamming
distance source for IVs.

3.1 ESP Environmental Considerations

Currently, there are no known issues regarding interactions between
these algorithms and other aspects of ESP, such as use of certain
authentication schemes.

3.2 Keying Material

The minimum number of bits sent from the key exchange protocol to
this ESP algorithm must be greater or equal to the key size.

The cipher's encryption and decryption key is taken from the first
<x> bits of the keying material, where <x> represents the required
key size.

4. Security Considerations

Implementations are encouraged to use the largest key sizes they can
when taking into account performance considerations for their
particular hardware and software configuration. Note that encryption
necessarily impacts both sides of a secure channel, so such
consideration must take into account not only the client side, but
the server as well.

For information on the case for using random values please see
[Bell97].

For further security considerations, the reader is encouraged to read
the documents that describe the actual cipher algorithms.

5. References

   [Adams97]   Adams, C, "The CAST-128 Encryption Algorithm",
                RFC2144, 1997.

   [Atkinson98]Kent, S. and R. Atkinson, "Security Architecture for the
                Internet Protocol", RFC 2401, November 1998.

   [Baldwin96] Baldwin, R. and R. Rivest, "The RC5, RC5-CBC, RC5-CBC-
                Pad, and RC5-CTS Algorithms", RFC 2040, October 1996.

   [Bell97]   S. Bellovin, "Probable Plaintext Cryptanalysis of the IP
                Security Protocols", Proceedings of the Symposium on
                Network and Distributed System Security, San Diego, CA,
                pp. 155-160, February 1997 (also
                http://www.research.att.com/˜smb/probtxt.{ps, pdf}).

   [Bosselaers]A. Bosselaers, "Performance of Pentium implementations",
                http://www.esat.kuleuven.ac.be/˜bosselae/

   [Bradner97] Bradner, S., "Key words for use in RFCs to indicate
                Requirement Levels", BCP 14, RFC 2119, March 1997.

   [Crypto93] J. Daemen, R. Govaerts, J. Vandewalle, "Weak Keys for
                IDEA", Advances in Cryptology, CRYPTO 93 Proceedings,
                Springer-Verlag, pp. 224-230.

   [FIPS-46]   US National Bureau of Standards, "Data Encryption
                Standard", Federal Information Processing Standard (FIPS)
                Publication 46, January 1977.

   [Kent98]   Kent, S. and R. Atkinson, "IP Encapsulating Security
                Payload (ESP)", RFC 2406, November 1998.

   [Lai]   X. Lai, "On the Design and Security of Block Ciphers",
                ETH Series in Information Processing, v. 1, Konstanz:
                Hartung-Gorre Verlag, 1992.

   [Madson98] Madson, C. and N. Dorswamy, "The ESP DES-CBC Cipher
                Algorithm With Explicit IV", RFC 2405, November 1998.

   [MOV]   A. Menezes, P. Van Oorschot, S. Vanstone, "Handbook of
                Applied Cryptography", CRC Press, 1997. ISBN 0-8493-
                8523-7

   [Schneier] B. Schneier, "Applied Cryptography Second Edition", John
                Wiley & Sons, New York, NY, 1995. ISBN 0-471-12845-7

   [Schneier93]B. Schneier, "Description of a New Variable-Length Key,
               64-Bit Block Cipher", from "Fast Software Encryption,
               Cambridge Security Workshop Proceedings", Springer-
               Verlag, 1994, pp. 191-204.
               http://www.counterpane.com/bfsverlag.html

   [Schneier95]B. Schneier, "The Blowfish Encryption Algorithm - One
               Year Later", Dr. Dobb's Journal, September 1995,
               http://www.counterpane.com/bfdobsoyl.html

   [Schneier97]B. Scheier, "Speed Comparisons of Block Ciphers on a
               Pentium." February 1997,
               http://www.counterpane.com/speed.html

   [Thayer97] Thayer, R., Doraswamy, N. and R. Glenn, "IP Security
               Document Roadmap", RFC 2411, November 1998.

   [Tuchman79] Tuchman, W, "Hellman Presents No Shortcut Solutions to
               DES", IEEE Spectrum, v. 16 n. 7, July 1979, pp. 40-41.

6. Acknowledgments

   This document is a merger of most of the ESP cipher algorithm
   documents. This merger was done to facilitate greater understanding
   of the commonality of all of the ESP algorithms and to further the
   development of these algorithm within ESP.

   The content of this document is based on suggestions originally from
   Stephen Kent and subsequent discussions from the IPSec mailing list
   as well as other IPSec documents.

   Special thanks to Carlisle Adams and Paul Van Oorschot both of
   Entrust Technologies who provided input and review of CAST.

   Thanks to all of the editors of the previous ESP 3DES documents; W.
   Simpson, N. Doraswamy, P. Metzger, and P. Karn.

   Thanks to Brett Howard from TimeStep for his original work of ESP-
   RC5.

   Thanks to Markku-Juhani Saarinen, Helger Lipmaa and Bart Preneel for
   their input on IDEA and other ciphers.

7. Editors' Addresses

   Roy Pereira
   TimeStep Corporation

   Phone: +1 (613) 599-3610 x 4808
   EMail: rpereira@timestep.com

   Rob Adams
   Cisco Systems Inc.

   Phone: +1 (408) 457-5397
   EMail: adams@cisco.com

   Contributors:

   Robert W. Baldwin
   RSA Data Security, Inc.

   Phone: +1 (415) 595-8782
   EMail: baldwin@rsa.com or baldwin@lcs.mit.edu

   Greg Carter
   Entrust Technologies

   Phone: +1 (613) 763-1358
   EMail: carterg@entrust.com

   Rodney Thayer
   Sable Technology Corporation

   Phone: +1 (617) 332-7292
   EMail: rodney@sabletech.com

The IPSec working group can be contacted via the IPSec working
group's mailing list (ipsec@tis.com) or through its chairs:

Robert Moskowitz
International Computer Security Association

EMail: rgm@icsa.net

Theodore Y. Ts'o
Massachusetts Institute of Technology

EMail: tytso@MIT.EDU

8. Full Copyright Statement