

The CAST-128 Encryption Algorithm

Status of this Memo

This memo provides information for the Internet community. This memo does not specify an Internet standard of any kind. Distribution of this memo is unlimited.

Abstract

There is a need in the Internet community for an unencumbered encryption algorithm with a range of key sizes that can provide security for a variety of cryptographic applications and protocols.

This document describes an existing algorithm that can be used to satisfy this requirement. Included are a description of the cipher and the key scheduling algorithm ([Section 2](#)), the s-boxes ([Appendix A](#)), and a set of test vectors ([Appendix B](#)).

TABLE OF CONTENTS

STATUS OF THIS MEMO.....	1
ABSTRACT.....	1
1. INTRODUCTION.....	1
2. DESCRIPTION OF ALGORITHM.....	2
3. INTELLECTUAL PROPERTY CONSIDERATIONS.....	8
4. SECURITY CONSIDERATIONS.....	8
5. REFERENCES.....	8
6. AUTHOR'S ADDRESS.....	8
APPENDICES	
A. S-BOXES.....	9
B. TEST VECTORS.....	15

1. Introduction

This document describes the CAST-128 encryption algorithm, a DES-like Substitution-Permutation Network (SPN) cryptosystem which appears to have good resistance to differential cryptanalysis, linear cryptanalysis, and related-key cryptanalysis. This cipher also possesses a number of other desirable cryptographic properties, including avalanche, Strict Avalanche Criterion (SAC), Bit Independence Criterion (BIC), no complementation property, and an absence of weak and semi-weak keys. It thus appears to be a good

candidate for general-purpose use throughout the Internet community wherever a cryptographically-strong, freely-available encryption algorithm is required.

Adams [[Adams](#)] discusses the CAST design procedure in some detail; analyses can also be obtained on-line (see, for example, [[Web1](#)] or [[Web2](#)]).

2. Description of Algorithm

CAST-128 belongs to the class of encryption algorithms known as Feistel ciphers; overall operation is thus similar to the Data Encryption Standard (DES). The full encryption algorithm is given in the following four steps.

INPUT: plaintext $m_1 \dots m_{64}$; key $K = k_1 \dots k_{128}$.

OUTPUT: ciphertext $c_1 \dots c_{64}$.

1. (key schedule) Compute 16 pairs of subkeys $\{K_{mi}, K_{ri}\}$ from K (see Sections 2.1 and 2.4).
2. $(L_0, R_0) \leftarrow (m_1 \dots m_{64})$. (Split the plaintext into left and right 32-bit halves $L_0 = m_1 \dots m_{32}$ and $R_0 = m_{33} \dots m_{64}$.)
3. (16 rounds) for i from 1 to 16, compute L_i and R_i as follows:
 $L_i = R_{i-1}$;
 $R_i = L_{i-1} \oplus f(R_{i-1}, K_{mi}, K_{ri})$, where f is defined in Section 2.2
 (f is of Type 1, Type 2, or Type 3, depending on i).
4. $c_1 \dots c_{64} \leftarrow (R_{16}, L_{16})$. (Exchange final blocks L_{16} , R_{16} and concatenate to form the ciphertext.)

Decryption is identical to the encryption algorithm given above, except that the rounds (and therefore the subkey pairs) are used in reverse order to compute (L_0, R_0) from (R_{16}, L_{16}) .

See [Appendix B](#) for test vectors which can be used to verify correctness of an implementation of this algorithm.

2.1. Pairs of Round Keys

CAST-128 uses a pair of subkeys per round: a 32-bit quantity K_m is used as a "masking" key and a 5-bit quantity K_r is used as a "rotation" key.

2.2. Non-Identical Rounds

Three different round functions are used in CAST-128. The rounds are as follows (where "D" is the data input to the f function and "Ia" - "Id" are the most significant byte through least significant byte of I, respectively). Note that "+" and "-" are addition and subtraction modulo $2^{*}32$, "^" is bitwise XOR, and "<<<" is the circular left-shift operation.

Type 1: $I = ((K_{mi} + D) \lll K_{ri})$
 $f = ((S1[Ia] \wedge S2[Ib]) - S3[Ic]) + S4[Id]$

Type 2: $I = ((K_{mi} \wedge D) \lll K_{ri})$
 $f = ((S1[Ia] - S2[Ib]) + S3[Ic]) \wedge S4[Id]$

Type 3: $I = ((K_{mi} - D) \lll K_{ri})$
 $f = ((S1[Ia] + S2[Ib]) \wedge S3[Ic]) - S4[Id]$

Rounds 1, 4, 7, 10, 13, and 16 use f function Type 1.

Rounds 2, 5, 8, 11, and 14 use f function Type 2.

Rounds 3, 6, 9, 12, and 15 use f function Type 3.

2.3. Substitution Boxes

CAST-128 uses eight substitution boxes: s-boxes S1, S2, S3, and S4 are round function s-boxes; S5, S6, S7, and S8 are key schedule s-boxes. Although 8 s-boxes require a total of 8 KBytes of storage, note that only 4 KBytes are required during actual encryption / decryption since subkey generation is typically done prior to any data input.

See [Appendix A](#) for the contents of s-boxes S1 - S8.

2.4. Key Schedule

Let the 128-bit key be $x_0x_1x_2x_3x_4x_5x_6x_7x_8x_9x_{Ax}x_{Bx}x_{Cx}x_{Dx}x_{Ex}x_{Fx}$, where x_0 represents the most significant byte and x_F represents the least significant byte.

Let $z_0..z_F$ be intermediate (temporary) bytes.

Let $S_i[]$ represent s-box i and let "^" represent XOR addition.

The subkeys are formed from the key $x_0x_1x_2x_3x_4x_5x_6x_7x_8x_9xAxBxCxDxExF$ as follows.

$$\begin{aligned} z_0z_1z_2z_3 &= x_0x_1x_2x_3 \wedge S5[xD] \wedge S6[xF] \wedge S7[xC] \wedge S8[xE] \wedge S7[x8] \\ z_4z_5z_6z_7 &= x_8x_9xAxB \wedge S5[z_0] \wedge S6[z_2] \wedge S7[z_1] \wedge S8[z_3] \wedge S8[xA] \\ z_8z_9zAzB &= xCxDxEzF \wedge S5[z_7] \wedge S6[z_6] \wedge S7[z_5] \wedge S8[z_4] \wedge S5[x_9] \\ zCzDzEzF &= x_4x_5x_6x_7 \wedge S5[zA] \wedge S6[z_9] \wedge S7[zB] \wedge S8[z_8] \wedge S6[xB] \\ K1 &= S5[z_8] \wedge S6[z_9] \wedge S7[z_7] \wedge S8[z_6] \wedge S5[z_2] \\ K2 &= S5[zA] \wedge S6[zB] \wedge S7[z_5] \wedge S8[z_4] \wedge S6[z_6] \\ K3 &= S5[zC] \wedge S6[zD] \wedge S7[z_3] \wedge S8[z_2] \wedge S7[z_9] \\ K4 &= S5[zE] \wedge S6[zF] \wedge S7[z_1] \wedge S8[z_0] \wedge S8[zC] \\ x_0x_1x_2x_3 &= z_8z_9zAzB \wedge S5[z_5] \wedge S6[z_7] \wedge S7[z_4] \wedge S8[z_6] \wedge S7[z_0] \\ x_4x_5x_6x_7 &= z_0z_1z_2z_3 \wedge S5[x_0] \wedge S6[x_2] \wedge S7[x_1] \wedge S8[x_3] \wedge S8[z_2] \\ x_8x_9xAxB &= z_4z_5z_6z_7 \wedge S5[x_7] \wedge S6[x_6] \wedge S7[x_5] \wedge S8[x_4] \wedge S5[z_1] \\ xCxDxEzF &= zCzDzEzF \wedge S5[xA] \wedge S6[x_9] \wedge S7[xB] \wedge S8[x_8] \wedge S6[z_3] \\ K5 &= S5[x_3] \wedge S6[x_2] \wedge S7[xC] \wedge S8[xD] \wedge S5[x_8] \\ K6 &= S5[x_1] \wedge S6[x_0] \wedge S7[xE] \wedge S8[xF] \wedge S6[xD] \\ K7 &= S5[x_7] \wedge S6[x_6] \wedge S7[x_8] \wedge S8[x_9] \wedge S7[x_3] \\ K8 &= S5[x_5] \wedge S6[x_4] \wedge S7[xA] \wedge S8[xB] \wedge S8[x_7] \\ z_0z_1z_2z_3 &= x_0x_1x_2x_3 \wedge S5[xD] \wedge S6[xF] \wedge S7[xC] \wedge S8[xE] \wedge S7[x_8] \\ z_4z_5z_6z_7 &= x_8x_9xAxB \wedge S5[z_0] \wedge S6[z_2] \wedge S7[z_1] \wedge S8[z_3] \wedge S8[xA] \\ z_8z_9zAzB &= xCxDxEzF \wedge S5[z_7] \wedge S6[z_6] \wedge S7[z_5] \wedge S8[z_4] \wedge S5[x_9] \\ zCzDzEzF &= x_4x_5x_6x_7 \wedge S5[zA] \wedge S6[z_9] \wedge S7[zB] \wedge S8[z_8] \wedge S6[xB] \\ K9 &= S5[z_3] \wedge S6[z_2] \wedge S7[zC] \wedge S8[zD] \wedge S5[z_9] \\ K10 &= S5[z_1] \wedge S6[z_0] \wedge S7[zE] \wedge S8[zF] \wedge S6[zC] \\ K11 &= S5[z_7] \wedge S6[z_6] \wedge S7[z_8] \wedge S8[z_9] \wedge S7[z_2] \\ K12 &= S5[z_5] \wedge S6[z_4] \wedge S7[zA] \wedge S8[zB] \wedge S8[z_6] \\ x_0x_1x_2x_3 &= z_8z_9zAzB \wedge S5[z_5] \wedge S6[z_7] \wedge S7[z_4] \wedge S8[z_6] \wedge S7[z_0] \\ x_4x_5x_6x_7 &= z_0z_1z_2z_3 \wedge S5[x_0] \wedge S6[x_2] \wedge S7[x_1] \wedge S8[x_3] \wedge S8[z_2] \\ x_8x_9xAxB &= z_4z_5z_6z_7 \wedge S5[x_7] \wedge S6[x_6] \wedge S7[x_5] \wedge S8[x_4] \wedge S5[z_1] \\ xCxDxEzF &= zCzDzEzF \wedge S5[xA] \wedge S6[x_9] \wedge S7[xB] \wedge S8[x_8] \wedge S6[z_3] \\ K13 &= S5[x_8] \wedge S6[x_9] \wedge S7[x_7] \wedge S8[x_6] \wedge S5[x_3] \\ K14 &= S5[xA] \wedge S6[xB] \wedge S7[x_5] \wedge S8[x_4] \wedge S6[x_7] \\ K15 &= S5[xC] \wedge S6[xD] \wedge S7[x_3] \wedge S8[x_2] \wedge S7[x_8] \\ K16 &= S5[xE] \wedge S6[xF] \wedge S7[x_1] \wedge S8[x_0] \wedge S8[xD] \end{aligned}$$

[The remaining half is identical to what is given above, carrying on from the last created $x0..xF$ to generate keys $K17 - K32$.]

```

z0z1z2z3 = x0x1x2x3 ^ S5[xD] ^ S6[xF] ^ S7[xC] ^ S8[xE] ^ S7[x8]
z4z5z6z7 = x8x9xAxB ^ S5[z0] ^ S6[z2] ^ S7[z1] ^ S8[z3] ^ S8[xA]
z8z9zAzB = xCxDxEzF ^ S5[z7] ^ S6[z6] ^ S7[z5] ^ S8[z4] ^ S5[x9]
zCzDzEzF = x4x5x6x7 ^ S5[zA] ^ S6[z9] ^ S7[zB] ^ S8[z8] ^ S6[xB]
K17 = S5[z8] ^ S6[z9] ^ S7[z7] ^ S8[z6] ^ S5[z2]
K18 = S5[zA] ^ S6[zB] ^ S7[z5] ^ S8[z4] ^ S6[z6]
K19 = S5[zC] ^ S6[zD] ^ S7[z3] ^ S8[z2] ^ S7[z9]
K20 = S5[zE] ^ S6[zF] ^ S7[z1] ^ S8[z0] ^ S8[zC]
x0x1x2x3 = z8z9zAzB ^ S5[z5] ^ S6[z7] ^ S7[z4] ^ S8[z6] ^ S7[z0]
x4x5x6x7 = z0z1z2z3 ^ S5[x0] ^ S6[x2] ^ S7[x1] ^ S8[x3] ^ S8[z2]
x8x9xAxB = z4z5z6z7 ^ S5[x7] ^ S6[x6] ^ S7[x5] ^ S8[x4] ^ S5[z1]
xCxDxEzF = zCzDzEzF ^ S5[xA] ^ S6[x9] ^ S7[xB] ^ S8[x8] ^ S6[z3]
K21 = S5[x3] ^ S6[x2] ^ S7[xC] ^ S8[xD] ^ S5[x8]
K22 = S5[x1] ^ S6[x0] ^ S7[xE] ^ S8[xF] ^ S6[xD]
K23 = S5[x7] ^ S6[x6] ^ S7[x8] ^ S8[x9] ^ S7[x3]
K24 = S5[x5] ^ S6[x4] ^ S7[xA] ^ S8[xB] ^ S8[x7]
z0z1z2z3 = x0x1x2x3 ^ S5[xD] ^ S6[xF] ^ S7[xC] ^ S8[xE] ^ S7[x8]
z4z5z6z7 = x8x9xAxB ^ S5[z0] ^ S6[z2] ^ S7[z1] ^ S8[z3] ^ S8[xA]
z8z9zAzB = xCxDxEzF ^ S5[z7] ^ S6[z6] ^ S7[z5] ^ S8[z4] ^ S5[x9]
zCzDzEzF = x4x5x6x7 ^ S5[zA] ^ S6[z9] ^ S7[zB] ^ S8[z8] ^ S6[xB]
K25 = S5[z3] ^ S6[z2] ^ S7[zC] ^ S8[zD] ^ S5[z9]
K26 = S5[z1] ^ S6[z0] ^ S7[zE] ^ S8[zF] ^ S6[zC]
K27 = S5[z7] ^ S6[z6] ^ S7[z8] ^ S8[z9] ^ S7[z2]
K28 = S5[z5] ^ S6[z4] ^ S7[zA] ^ S8[zB] ^ S8[z6]
x0x1x2x3 = z8z9zAzB ^ S5[z5] ^ S6[z7] ^ S7[z4] ^ S8[z6] ^ S7[z0]
x4x5x6x7 = z0z1z2z3 ^ S5[x0] ^ S6[x2] ^ S7[x1] ^ S8[x3] ^ S8[z2]
x8x9xAxB = z4z5z6z7 ^ S5[x7] ^ S6[x6] ^ S7[x5] ^ S8[x4] ^ S5[z1]
xCxDxEzF = zCzDzEzF ^ S5[xA] ^ S6[x9] ^ S7[xB] ^ S8[x8] ^ S6[z3]
K29 = S5[x8] ^ S6[x9] ^ S7[x7] ^ S8[x6] ^ S5[x3]
K30 = S5[xA] ^ S6[xB] ^ S7[x5] ^ S8[x4] ^ S6[x7]
K31 = S5[xC] ^ S6[xD] ^ S7[x3] ^ S8[x2] ^ S7[x8]
K32 = S5[xE] ^ S6[xF] ^ S7[x1] ^ S8[x0] ^ S8[xD]

```

2.4.1. Masking Subkeys And Rotate Subkeys

Let $Km1, \dots, Km16$ be 32-bit masking subkeys (one per round).
 Let $Kr1, \dots, Kr16$ be 32-bit rotate subkeys (one per round); only the least significant 5 bits are used in each round.

```
for (i=1; i<=16; i++) { Kmi = Ki; Kri = K16+i; }
```

2.5. Variable Keysize

The CAST-128 encryption algorithm has been designed to allow a key size that can vary from 40 bits to 128 bits, in 8-bit increments (that is, the allowable key sizes are 40, 48, 56, 64, ..., 112, 120, and 128 bits. For variable keysize operation, the specification is as follows:

- 1) For key sizes up to and including 80 bits (i.e., 40, 48, 56, 64, 72, and 80 bits), the algorithm is exactly as specified but uses 12 rounds instead of 16;
- 2) For key sizes greater than 80 bits, the algorithm uses the full 16 rounds;
- 3) For key sizes less than 128 bits, the key is padded with zero bytes (in the rightmost, or least significant, positions) out to 128 bits (since the CAST-128 key schedule assumes an input key of 128 bits).

Note that although CAST-128 can support all 12 key sizes listed above, 40 bits, 64 bits, 80 bits, and 128 bits are the sizes that find utility in typical environments. Therefore, it will likely be sufficient for most implementations to support some subset of only these four sizes.

In order to avoid confusion when variable keysize operation is used, the name CAST-128 is to be considered synonymous with the name CAST5; this allows a keysize to be appended without ambiguity. Thus, for example, CAST-128 with a 40-bit key is to be referred to as CAST5-40; where a 128-bit key is explicitly intended, the name CAST5-128 should be used.

2.6. CAST5 Object Identifiers

For those who may be using CAST in algorithm negotiation within a protocol, or in any other context which may require the use of OBJECT IDENTIFIERS, the following OIDs have been defined.

```
algorithms OBJECT IDENTIFIER ::=
  { iso(1) memberBody(2) usa(840) nt(113533) nsn(7) algorithms(66) }
```

cast5CBC OBJECT IDENTIFIER ::= { algorithms cast5CBC(10) }

```
Parameters ::= SEQUENCE {
    iv          OCTET STRING DEFAULT 0,  -- Initialization vector
    keyLength   INTEGER                  -- Key length, in bits
}
```

Note: The iv is optional and defaults to all-zero. On the encoding end, if an all-zero iv is used, then it should be absent from the Parameters. On the decoding end, an absent iv should be interpreted as meaning all-zeros.

This is encryption and decryption in CBC mode using the CAST-128 symmetric block cipher algorithm.

cast5MAC OBJECT IDENTIFIER ::= { algorithms cast5MAC(11) }

```
Parameters ::= SEQUENCE {
    macLength   INTEGER,                -- MAC length, in bits
    keyLength   INTEGER                  -- Key length, in bits
}
```

This is message authentication using the CAST-128 symmetric block cipher algorithm.

pbeWithMD5AndCast5CBC OBJECT IDENTIFIER ::= { algorithms pbeWithMD5AndCAST5-CBC(12) }

```
Parameters ::= SEQUENCE {
    salt          OCTET STRING,
    iterationCount INTEGER,            -- Total number of hash iterations
    keyLength     INTEGER              -- Key length, in bits
}
```

Note: The IV is derived from the hashing procedure and therefore need not be included in Parameters.

This is password-based encryption and decryption in CBC mode using MD5 and the CAST-128 symmetric block cipher. See PKCS #5 (which uses the DES cipher) for details of the PBE computation.

2.7. Discussion

CAST-128 is a 12- or 16-round Feistel cipher that has a blocksize of 64 bits and a keysize of up to 128 bits; it uses rotation to provide intrinsic immunity to linear and differential attacks; it uses a mixture of XOR, addition and subtraction (modulo $2^{*}32$) in the round function; and it uses three variations of the round function itself throughout the cipher. Finally, the 8×32 s-boxes used in the round function each have a minimum nonlinearity of 74 and a maximum entry of 2 in the difference distribution table.

This cipher appears to have cryptographic strength in accordance with its keysize (128 bits) and has very good encryption / decryption performance: 3.3 MBytes/sec on a 150 MHz Pentium processor.

3. Intellectual Property Considerations

The CAST-128 cipher described in this document is available worldwide on a royalty-free basis for commercial and non-commercial uses.

4. Security Considerations

This entire memo is about security since it describes an algorithm which is specifically intended for cryptographic purposes.

5. References

[Adams] Adams, C., "Constructing Symmetric Ciphers using the CAST Design Procedure", Designs, Codes, and Cryptography (to appear).

[Web1] "Constructing Symmetric Ciphers using the CAST Design Procedure" (identical to [Adams] but available on-line) and "CAST Design Procedure Addendum", <http://www.entrust.com/library.htm>.

[Web2] "CAST Encryption Algorithm Related Publications", <http://adonis.ee.queensu.ca:8000/cast/cast.html>.

6. Author's Address

Carlisle Adams
Entrust Technologies
750 Heron Road,
Ottawa, Canada, K1V 1A7

E-mail: cadams@entrust.com
Phone: +1.613.763.9008

Appendix A. S-Boxes

S-Box S1

30fb40d4	9fa0ff0b	6beccd2f	3f258c7a	1e213f2f	9c004dd3	6003e540	cf9fc949
bfd4af27	88bbdb5	e2034090	98d09675	6e63a0e0	15c361d2	c2e7661d	22d4ff8e
28683b6f	c07fd059	ff2379c8	775f50e2	43c340d3	df2f8656	887ca41a	a2d2bd2d
alc9e0d6	346c4819	61b76d87	22540f2f	2abe32e1	aa54166b	22568e3a	a2d341d0
66db40c8	a784392f	004dff2f	2db9d2de	97943fac	4a97c1d8	527644b7	b5f437a7
b82cbaef	d751d159	6ff7f0ed	5a097a1f	827b68d0	90ecf52e	22b0c054	bc8e5935
4b6d2f7f	50bb64a2	d2664910	bee5812d	b7332290	e93b159f	b48ee411	4bff345d
fd45c240	ad31973f	c4f6d02e	55fc8165	d5b1caad	alac2dae	a2d4b76d	c19b0c50
882240f2	0c6e4f38	a4e4bfd7	4f5ba272	564c1d2f	c59c5319	b949e354	b04669fe
b1b6ab8a	c71358dd	6385c545	110f935d	57538ad5	6a390493	e63d37e0	2a54f6b3
3a787d5f	6276a0b5	19a6fcdf	7a42206a	29f9d4d5	f61b1891	bb72275e	aa508167
38901091	c6b505eb	84c7cb8c	2ad75a0f	874a1427	a2d1936b	2ad286af	aa56d291
d7894360	425c750d	93b39e26	187184c9	6c00b32d	73e2bb14	a0bebc3c	54623779
64459eab	3f328b82	7718cf82	59a2cea6	04ee002e	89fe78e6	3fab0950	325ff6c2
81383f05	6963c5c8	76cb5ad6	d49974c9	ca180dcf	380782d5	c7fa5cf6	8ac31511
35e79e13	47da91d0	f40f9086	a7e2419e	31366241	051ef495	aa573b04	4a805d8d
548300d0	00322a3c	bf64cddf	ba57a68e	75c6372b	50afd341	a7c13275	915a0bf5
6b54bfab	2b0b1426	ab4cc9d7	449ccd82	f7fbf265	ab85c5f3	1b55db94	aad4e324
cfa4bd3f	2deaa3e2	9e204d02	c8bd25ac	eadf55b3	d5bd9e98	e31231b2	2ad5ad6c
954329de	adbe4528	d8710f69	aa51c90f	aa786bf6	22513f1e	aa51a79b	2ad344cc
7b5a41f0	d37cfbad	1b069505	41ece491	b4c332e6	032268d4	c9600acc	ce387e6d
bf6bb16c	6a70fb78	0d03d9c9	d4df39de	e01063da	4736f464	5ad328d8	b347cc96
75bb0fc3	98511bfb	4ffbcc35	b58bcf6a	e11f0abc	bfc5fe4a	a70aec10	ac39570a
3f04442f	6188b153	e0397a2e	5727cb79	9ceb418f	1cacd68d	2ad37c96	0175cb9d
c69dff09	c75b65f0	d9db40d8	ec0e7779	4744ead4	b11c3274	dd24cb9e	7e1c54bd
f01144f9	d2240eb1	9675b3fd	a3ac3755	d47c27af	51c85f4d	56907596	a5bb15e6
580304f0	ca042cf1	011a37ea	8dbfaadb	35ba3e4a	3526ffa0	c37b4d09	bc306ed9
98a52666	5648f725	ff5e569d	0ced63d0	7c63b2cf	700b45e1	d5ea50f1	85a92872
af1fbdad	d4234870	a7870bf3	2d3b4d79	42e04198	0cd0ede7	26470db8	f881814c
474d6ad7	7c0c5e5c	d1231959	381b7298	f5d2f4db	ab838653	6e2f1e23	83719c9e
bd91e046	9a56456e	dc39200c	20c8c571	962bdalc	e1e696ff	b141ab08	7cca89b9
1a69e783	02cc4843	a2f7c579	429ef47d	427b169c	5ac9f049	dd8f0f00	5c8165bf

S-Box S2

1f201094	ef0ba75b	69e3cf7e	393f4380	fe61cf7a	eec5207a	55889c94	72fc0651
ada7ef79	4e1d7235	d55a63ce	de0436ba	99c430ef	5f0c0794	18dcdb7d	a1d6eff3
a0b52f7b	59e83605	ee15b094	e9ffd909	dc440086	ef944459	ba83ccb3	e0c3cdfb
d1da4181	3b092ab1	f997f1c1	a5e6cf7b	01420ddb	e4e7ef5b	25a1ff41	e180f806
1fc41080	179bee7a	d37ac6a9	fe5830a4	98de8b7f	77e83f4e	79929269	24fa9f7b
e113c85b	acc40083	d7503525	f7ea615f	62143154	0d554b63	5d681121	c866c359
3d63cf73	cee234c0	d4d87e87	5c672b21	071f6181	39f7627f	361e3084	e4eb573b
602f64a4	d63acd9c	1bbc4635	9e81032d	2701f50c	99847ab4	a0e3df79	ba6cf38c
10843094	2537a95e	f46f6ffe	a1ff3b1f	208cfb6a	8f458c74	d9e0a227	4ec73a34
fc884f69	3e4de8df	ef0e0088	3559648d	8a45388c	1d804366	721d9bfd	a58684bb
e8256333	844e8212	128d8098	fed33fb4	ce280ae1	27e19ba5	d5a6c252	e49754bd

c5d655dd	eb667064	77840b4d	a1b6a801	84db26a9	e0b56714	21f043b7	e5d05860
54f03084	066ff472	a31aa153	dadc4755	b5625dbf	68561be6	83ca6b94	2d6ed23b
eccf01db	a6d3d0ba	b6803d5c	af77a709	33b4a34c	397bc8d6	5ee22b95	5f0e5304
81ed6f61	20e74364	b45e1378	de18639b	881ca122	b96726d1	8049a7e8	22b7da7b
5e552d25	5272d237	79d2951c	c60d894c	488cb402	1ba4fe5b	a4b09f6b	1ca815cf
a20c3005	8871df63	b9de2fcb	0cc6c9e9	0beeff53	e3214517	b4542835	9f63293c
ee41e729	6e1d2d7c	50045286	1e6685f3	f33401c6	30a22c95	31a70850	60930f13
73f98417	a1269859	ec645c44	52c877a9	cdff33a6	a02b1741	7cbad9a2	2180036f
50d99c08	cb3f4861	c26bd765	64a3f6ab	80342676	25a75e7b	e4e6d1fc	20c710e6
cdf0b680	17844d3b	31eef84d	7e0824e4	2ccb49eb	846a3bae	8ff77888	ee5d60f6
7af75673	2fdd5cdb	a11631c1	30f66f43	b3faec54	157fd7fa	ef8579cc	d152de58
db2fffd5e	8f32ce19	306af97a	02f03ef8	99319ad5	c242fa0f	a7e3ebb0	c68e4906
b8da230c	80823028	dcdef3c8	d35fb171	088a1bc8	bec0c560	61a3c9e8	bca8f54d
c72feffa	22822e99	82c570b4	d8d94e89	8b1c34bc	301e16e6	273be979	b0ffea6
61d9b8c6	00b24869	b7ffce3f	08dc283b	43daf65a	f7e19798	7619b72f	8f1c9ba4
dc8637a0	16a7d3b1	9fc393b7	a7136eeb	c6bcc63e	1a513742	ef6828bc	520365d6
2d6a77ab	3527ed4b	821fd216	095c6e2e	db92f2fb	5eea29cb	145892f5	91584f7f
5483697b	2667a8cc	85196048	8c4bacea	833860d4	0d23e0f9	6c387e8a	0ae6d249
b284600c	d835731d	dcb1c647	ac4c56ea	3ebd81b3	230eabb0	6438bc87	f0b5b1fa
8f5ea2b3	fc184642	0a036b7a	4fb089bd	649da589	a345415e	5c038323	3e5d3bb9
43d79572	7e6dd07c	06dfdf1e	6c6cc4ef	7160a539	73bfbe70	83877605	4523ecf1

S-Box S3

8defc240	25fa5d9f	eb903dbf	e810c907	47607fff	369fe44b	8c1fc644	aececa90
beb1f9bf	eeefbcae	e8cf1950	51df07ae	920e8806	f0ad0548	e13c8d83	927010d5
11107d9f	07647db9	b2e3e4d4	3d4f285e	b9afa820	fade82e0	a067268b	8272792e
553fb2c0	489ae22b	d4ef9794	125e3fbc	21fffcee	825b1bfd	9255c5ed	1257a240
4e1a8302	bae07fff	528246e7	8e57140e	3373f7bf	8c9f8188	a6fc4ee8	c982b5a5
a8c01db7	579fc264	67094f31	f2bd3f5f	40fff7c1	1fb78dfc	8e6bd2c1	437be59b
99b03dbf	b5dbc64b	638dc0e6	55819d99	a197c81c	4a012d6e	c5884a28	ccc36f71
b843c213	6c0743f1	8309893c	0feddd5f	2f7fe850	d7c07f7e	02507fbf	5afb9a04
a747d2d0	1651192e	af70bf3e	58c31380	5f98302e	727cc3c4	0a0fb402	0f7fef82
8c96fdad	5d2c2aae	8ee99a49	50da88b8	8427f4a0	1eac5790	796fb449	8252dc15
efbd7d9b	a672597d	ada840d8	45f54504	fa5d7403	e83ec305	4f91751a	925669c2
23efe941	a903f12e	60270df2	0276e4b6	94fd6574	927985b2	8276dbcb	02778176
f8af918d	4e48f79e	8f616ddf	e29d840e	842f7d83	340ce5c8	96bbb682	93b4b148
ef303cab	984faf28	779faf9b	92dc560d	224d1e20	8437aa88	7d29dc96	2756d3dc
8b907cee	b51fd240	e7c07ce3	e566b4a1	c3e9615e	3cf8209d	6094d1e3	cd9ca341
5c76460e	00ea983b	d4d67881	fd47572c	f76cedd9	bda8229c	127dadaa	438a074e
1f97c090	081bdb8a	93a07ebe	b938ca15	97b03cff	3dc2c0f8	8d1ab2ec	64380e51
68cc7bfb	d90f2788	12490181	5de5ffd4	dd7ef86a	76a2e214	b9a40368	925d958f
4b39fffa	ba39aee9	a4ffd30b	faf7933b	6d498623	193cbcfa	27627545	825cf47a
61bd8ba0	d11e42d1	cead04f4	127ea392	10428db7	8272a972	9270c4a8	127de50b
285balc8	3c62f44f	35c0eaa5	e805d231	428929fb	b4fcd82	4fb66a53	0e7dc15b
1f081fab	108618ae	fcfd086d	f9ff2889	694bcc11	236a5cae	12deca4d	2c3f8cc5
d2d02dfe	f8ef5896	e4cf52da	95155b67	494a488c	b9b6a80c	5c8f82bc	89d36b45
3a609437	ec00c9a9	44715253	0a874b49	d773bc40	7c34671c	02717ef6	4feb5536
a2d02fff	d2bf60c4	d43f03c0	50b4ef6d	07478cd1	006e1888	a2e53f55	b9e6d4bc

```

a2048016 97573833 d7207d67 de0f8f3d 72f87b33 abcc4f33 7688c55d 7b00a6b0
947b0001 570075d2 f9bb88f8 8942019e 4264a5ff 856302e0 72dbd92b ee971b69
6ea22fde 5f08ae2b af7a616d e5c98767 cf1febd2 61efc8c2 flac2571 cc8239c2
67214cb8 b1e583d1 b7dc3e62 7f10bdce f90a5c38 0ff0443d 606e6dc6 60543a49
5727c148 2be98a1d 8ab41738 20e1be24 af96da0f 68458425 99833be5 600d457d
282f9350 8334b362 d91d1120 2b6d8da0 642b1e31 9c305a00 52bce688 1b03588a
f7baefd5 4142ed9c a4315c11 83323ec5 dfef4636 a133c501 e9d3531c ee353783

```

S-Box S4

```

9db30420 1fb6e9de a7be7bef d273a298 4a4f7bdb 64ad8c57 85510443 fa020ed1
7e287aff e60fb663 095f35a1 79ebf120 fd059d43 6497b7b1 f3641f63 241e4adf
28147f5f 4fa2b8cd c9430040 0cc32220 fdd30b30 c0a5374f 1d2d00d9 24147b15
ee4d111a 0fca5167 71ff904c 2d195ffe 1a05645f 0c13fefe 081b08ca 05170121
80530100 e83e5efe ac9af4f8 7fe72701 d2b8ee5f 06df4261 bb9e9b8a 7293ea25
ce84ffdf f5718801 3dd64b04 a26f263b 7ed48400 547eebe6 446d4ca0 6cf3d6f5
2649abdf aea0c7f5 36338cc1 503f7e93 d3772061 11b638e1 72500e03 f80eb2bb
abe0502e ec8d77de 57971e81 e14f6746 c9335400 6920318f 081dbb99 ffc304a5
4d351805 7f3d5ce3 a6c866c6 5d5bcc9a daec6fea 9f926f91 9f46222f 3991467d
a5bf6d8e 1143c44f 43958302 d0214eeb 022083b8 3fb6180c 18f8931e 281658e6
26486e3e 8bd78a70 7477e4c1 b506e07c f32d0a25 79098b02 e4eabb81 28123b23
69dead38 1574ca16 df871b62 211c40b7 a51a9ef9 0014377b 041e8ac8 09114003
bd59e4d2 e3d156d5 4fe876d5 2f91a340 557be8de 00eae4a7 0ce5c2ec 4db4bba6
e756bdf5 dd3369ac ec17b035 06572327 99afc8b0 56c8c391 6b65811c 5e146119
6e85cb75 be07c002 c2325577 893ff4ec 5bbfc92d d0ec3b25 b7801ab7 8d6d3b24
20c763ef c366a5fc 9c382880 0ace3205 aac9548a ecald7c7 041afa32 1d16625a
6701902c 9b757a54 31d477f7 9126b031 36cc6fdb c70b8b46 d9e66a48 56e55a79
026a4ceb 52437eff 2f8f76b4 0df980a5 8674cde3 edda04eb 17a9be04 2c18f4df
b7747f9d ab2af7b4 efc34d20 2e096b7c 1741a254 e5b6a035 213d42f6 2c1c7c26
61c2f50f 6552daf9 d2c231f8 25130f69 d8167fa2 0418f2c8 001a96a6 0d1526ab
63315c21 5e0a72ec 49bafefd 187908d9 8d0dbd86 311170a7 3e9b640c cc3e10d7
d5cad3b6 0caec388 f73001e1 6c728aff 71eae2a1 1f9af36e cfcabd12f c1de8417
ac07be6b cb44a1d8 8b9b0f56 013988c3 b1c52fca b4be31cd d8782806 12a3a4e2
6f7de532 58fd7eb6 d01ee900 24adffc2 f4990fc5 9711aac5 001d7b95 82e5e7d2
109873f6 00613096 c32d9521 ada121ff 29908415 7fbb977f af9eb3db 29c9ed2a
5ce2a465 a730f32c d0aa3fe8 8a5cc091 d49e2ce7 0ce454a9 d60acd86 015f1919
77079103 dea03af6 78a8565e dee356df 21f05cbe 8b75e387 b3c50651 b8a5c3ef
d8eeb6d2 e523be77 c2154529 2f69efdf afe67afb f470c4b2 f3e0eb5b d6cc9876
39e4460c 1fda8538 1987832f ca007367 a99144f8 296b299e 492fc295 9266beab
b5676e69 9bd3ddda df7e052f db25701c 1b5e51ee f65324e6 6afce36c 0316cc04
8644213e b7dc59d0 7965291f ccd6fd43 41823979 932bcd6f b657c34d 4edfd282
7ae5290c 3cb9536b 851e20fe 9833557e 13ecf0b0 d3ffb372 3f85c5c1 0aef7ed2

```

S-Box S5

```

7ec90c04 2c6e74b9 9b0e66df a6337911 b86a7fff 1dd358f5 44dd9d44 1731167f
08fbf1fa e7f511cc d2051b00 735aba00 2ab722d8 386381cb acf6243a 69befd7a
e6a2e77f f0c720cd c4494816 ccf5c180 38851640 15b0a848 e68b18cb 4caadef5
5f480a01 0412b2aa 259814fc 41d0efe2 4e40b48d 248eb6fb 8dbalcfe 41a99b02
1a550a04 ba8f65cb 7251f4e7 95a51725 c106ecd7 97a5980a c539b9aa 4d79fe6a

```

f2f3f763	68af8040	ed0c9e56	11b4958b	e1eb5a88	8709e6b0	d7e07156	4e29fea7
6366e52d	02d1c000	c4ac8e05	9377f571	0c05372a	578535f2	2261be02	d642a0c9
df13a280	74b55bd2	682199c0	d421e5ec	53fb3ce8	c8adedb3	28a87fc9	3d959981
5c1ff900	fe38d399	0c4eff0b	062407ea	aa2f4fb1	4fb96976	90c79505	b0a8a774
ef55a1ff	e59ca2c2	a6b62d27	e66a4263	df65001f	0ec50966	dfdd55bc	29de0655
911e739a	17af8975	32c7911c	89f89468	0d01e980	524755f4	03b63cc9	0cc844b2
bcf3f0aa	87ac36e9	e53a7426	01b3d82b	1a9e7449	64ee2d7e	cddbbl1da	01c94910
b868bf80	0d26f3fd	9342ede7	04a5c284	636737b6	50f5b616	f24766e3	8eca36c1
136e05db	fef18391	fb887a37	d6e7f7d4	c7fb7dc9	3063fcd	b6f589de	ec2941da
26e46695	b7566419	f654efc5	d08d58b7	48925401	c1bacb7f	e5ff550f	b6083049
5bb5d0e8	87d72e5a	ab6a6ee1	223a66ce	c62bf3cd	9e0885f9	68cb3e47	086c010f
a21de820	d18b69de	f3f65777	fa02c3f6	407edac3	cbb3d550	1793084d	b0d70eba
0ab378d5	d951fb0c	ded7da56	4124bbe4	94ca0b56	0f5755d1	e0e1e56e	6184b5be
580a249f	94f74bc0	e327888e	9f7b5561	c3dc0280	05687715	646c6bd7	44904db3
66b4f0a3	c0f1648a	697ed5af	49e92ff6	309e374f	2cb6356a	85808573	4991f840
76f0ae02	083be84d	28421c9a	44489406	736e4cb8	c1092910	8bc95fc6	7d869cf4
134f616f	2e77118d	b31b2be1	aa90b472	3ca5d717	7d161bba	9cad9010	af462ba2
9fe459d2	45d34559	d9f2da13	dbc65487	f3e4f94e	176d486f	097c13ea	631da5c7
445f7382	175683f4	cdc66a97	70be0288	b3cdcf72	6e5dd2f3	20936079	459b80a5
be60e2db	a9c23101	eba5315c	224e42f2	1c5c1572	f6721b2c	1ad2fff3	8c25404e
324ed72f	4067b7fd	0523138e	5ca3bc78	dc0fd66e	75922283	784d6b17	58ebb16e
44094f85	3f481d87	fcfeae7b	77b5fff7	8c2302bf	aaf47556	5f46b02a	2b092801
3d38f5f7	0ca81f36	52af4a8a	66d5e7c0	df3b0874	95055110	1b5ad7a8	f61ed5ad
6cf6e479	20758184	d0cefa65	88f7be58	4a046826	0ff6f8f3	a09c7f70	5346aba0
5ce96c28	e176eda3	6bac307f	376829d2	85360fa9	17e3fe2a	24b79767	f5a96b20
d6cd2595	68ff1ebf	7555442c	f19f06be	f9e0659a	eeb9491d	34010718	bb30cab8
e822fe15	88570983	750e6249	da627e55	5e76ffa8	b1534546	6d47de08	efe9e7d4

S-Box S6

f6fa8f9d	2cac6ce1	4ca34867	e2337f7c	95db08e7	016843b4	eced5cbc	325553ac
bf9f0960	dfa1e2ed	83f0579d	63ed86b9	1ab6a6b8	de5ebe39	f38ff732	8989b138
33f14961	c01937bd	f506c6da	e4625e7e	a308ea99	4e23e33c	79cbd7cc	48a14367
a3149619	fec94bd5	a114174a	eea01866	a084db2d	09a8486f	a888614a	2900af98
01665991	e1992863	c8f30c60	2e78ef3c	d0d51932	cf0fec14	f7ca07d2	d0a82072
fd41197e	9305a6b0	e86be3da	74bed3cd	372da53c	4c7f4448	dab5d440	6dba0ec3
083919a7	9fbaeed9	49dbcfb0	4e670c53	5c3d9c01	64bdb941	2c0e636a	ba7dd9cd
ea6f7388	e70bc762	35f29adb	5c4cdd8d	f0d48d8c	b88153e2	08a19866	1ae2eac8
284caf89	aa928223	9334be53	3b3a21bf	16434be3	9aea3906	efe8c36e	f890cdd9
80226dae	c340a4a3	df7e9c09	a694a807	5b7c5ecc	221db3a6	9a69a02f	68818a54
ceb2296f	53c0843a	fe893655	25bfe68a	b4628abc	cf222ebf	25ac6f48	a9a99387
53bddb65	e76ffbe7	e967fd78	0ba93563	8e342bc1	e8a11be9	4980740d	c8087dfc
8de4bf99	a11101a0	7fd37975	da5a26c0	e81f994f	9528cd89	fd339fed	b87834bf
5f04456d	22258698	c9c4c83b	2dc156be	4f628daa	57f55ec5	e2220abe	d2916ebf
4ec75b95	24f2c3c0	42d15d99	cd0d7fa0	7b6e27ff	a8dc8af0	7345c106	f41e232f
35162386	e6ea8926	3333b094	157ec6f2	372b74af	692573e4	e9a9d848	f3160289
3a62ef1d	a787e238	f3a5f676	74364853	20951063	4576698d	b6fad407	592af950
36f73523	4cfb6e87	7da4cec0	6c152daa	cb0396a8	c50dfe5d	fcd707ab	0921c42f
89dff0bb	5fe2be78	448f4f33	754613c9	2b05d08d	48b9d585	dc049441	c8098f9b

7dede786	c39a3373	42410005	6a091751	0ef3c8a6	890072d6	28207682	a9a9f7be
bf32679d	d45b5b75	b353fd00	cbb0e358	830f220a	1f8fb214	d372cf08	cc3c4a13
8cf63166	061c87be	88c98f88	6062e397	47cf8e7a	b6c85283	3cc2acfb	3fc06976
4e8f0252	64d8314d	da3870e3	1e665459	c10908f0	513021a5	6c5b68b7	822f8aa0
3007cd3e	74719eef	dc872681	073340d4	7e432fd9	0c5ec241	8809286c	f592d891
08a930f6	957ef305	b7fbffbd	c266e96f	6fe4ac98	b173ecc0	bc60b42a	953498da
fbalae12	2d4bd736	0f25faab	a4f3fceb	e2969123	257f0c3d	9348af49	361400bc
e8816f4a	3814f200	a3f94043	9c7a54c2	bc704f57	da41e7f9	c25ad33a	54f4a084
b17f5505	59357cbe	edbd15c8	7f97c5ab	ba5ac7b5	b6f6deaf	3a479c3a	5302da25
653d7e6a	54268d49	51a477ea	5017d55b	d7d25d88	44136c76	0404a8c8	b8e5a121
b81a928a	60ed5869	97c55b96	eaec991b	29935913	01fdb7f1	088e8dfa	9ab6f6f5
3b4cbf9f	4a5de3ab	e6051d35	a0e1d855	d36b4cf1	f544edeb	b0e93524	bebb8fbdb
a2d762cf	49c92f54	38b5f331	7128a454	48392905	a65b1db8	851c97bd	d675cf2f

S-Box S7

85e04019	332bf567	662dbfff	cfc65693	2a8d7f6f	ab9bc912	de6008a1	2028da1f
0227bce7	4d642916	18fac300	50f18b82	2cb2cb11	b232e75c	4b3695f2	b28707de
a05fbcf6	cd4181e9	e150210c	e24ef1bd	b168c381	fde4e789	5c79b0d8	1e8bfd43
4d495001	38be4341	913cee1d	92a79c3f	089766be	baeeadf4	1286becf	b6eacb19
2660c200	7565bde4	64241f7a	8248dca9	c3b3ad66	28136086	0bd8dfa8	356d1cf2
107789be	b3b2e9ce	0502aa8f	0bc0351e	166bf52a	eb12ff82	e3486911	d34d7516
4e7b3aff	5f43671b	9cf6e037	4981ac83	334266ce	8c9341b7	d0d854c0	cb3a6c88
47bc2829	4725ba37	a66ad22b	7ad61f1e	0c5cbafa	4437f107	b6e79962	42d2d816
0a961288	e1a5c06e	13749e67	72fc081a	b1d139f7	f9583745	cf19df58	bec3f756
c06eba30	07211b24	45c28829	c95e317f	bc8ec511	38bc46e9	c6e6fa14	bae8584a
ad4ebc46	468f508b	7829435f	f124183b	821dba9f	aff60ff4	ea2c4e6d	16e39264
92544a8b	009b4fc3	aba68ced	9ac96f78	06a5b79a	b2856e6e	1aec3ca9	be838688
0e0804e9	55f1be56	e7e5363b	b3a1f25d	f7debb85	61fe033c	16746233	3c034c28
da6d0c74	79aac56c	3ce4e1ad	51f0c802	98f8f35a	1626a49f	eed82b29	1d382fe3
0c4fb99a	bb325778	3ec6d97b	6e77a6a9	cb658b5c	d45230c7	2bd1408b	60c03eb7
b9068d78	a33754f4	f430c87d	c8a71302	b96d8c32	ebd4e7be	be8b9d2d	7979fb06
e7225308	8b75cf77	11ef8da4	e083c858	8d6b786f	5a6317a6	fa5cf7a0	5dda0033
f28ebfb0	f5b9c310	a0eac280	08b9767a	a3d9d2b0	79d34217	021a718d	9ac6336a
2711fd60	438050e3	069908a8	3d7fedc4	826d2bef	4eeb8476	488dcf25	36c9d566
28e74e41	c2610aca	3d49a9cf	bae3b9df	b65f8de6	92aeaf64	3ac7d5e6	9ea80509
f22b017d	a4173f70	dd1e16c3	15e0d7f9	50b1b887	2b9f4fd5	625aba82	6a017962
2ec01b9c	15488aa9	d716e740	40055a2c	93d29a22	e32dbf9a	058745b9	3453dc1e
d699296e	496cff6f	1c9f4986	dfe2ed07	b87242d1	19de7eae	053e561a	15ad6f8c
66626c1c	7154c24c	ea082b2a	93eb2939	17dcb0f0	58d4f2ae	9ea294fb	52cf564c
9883fe66	2ec40581	763953c3	01d6692e	d3a0c108	a1e7160e	e4f2dfa6	693ed285
74904698	4c2b0edd	4f757656	5d393378	a132234f	3d321c5d	c3f5e194	4b269301
c79f022f	3c997e7e	5e4f9504	3ffaafbbd	76f7ad0e	296693f4	3d1fce6f	c61e45be
d3b5ab34	f72bf9b7	1b0434c0	4e72b567	5592a33d	b5229301	cf2a87f	60aeb767
1814386b	30bcc33d	38a0c07d	fd1606f2	c363519b	589dd390	5479f8e6	1cb8d647
97fd61a9	ea7759f4	2d57539d	569a58cf	e84e63ad	462e1b78	6580f87e	f3817914
91da55f4	40a230f3	d1988f35	b6e318d2	3ffa50bc	3d40f021	c3c0bdae	4958c24c
518f36b2	84b1d370	0fedce83	878ddada	f2a279c7	94e01be8	90716f4b	954b8aa3

S-Box S8

e216300d	bbddfffc	a7ebdabd	35648095	7789f8b7	e6c1121b	0e241600	052ce8b5
11a9cfb0	e5952f11	ece7990a	9386d174	2a42931c	76e38111	b12def3a	37dddfc
de9adeb1	0a0cc32c	be197029	84a00940	bb243a0f	b4d137cf	b44e79f0	049eedfd
0b15a15d	480d3168	8bbbde5a	669ded42	c7ece831	3f8f95e7	72df191b	7580330d
94074251	5c7dcdfa	abbe6d63	aa402164	b301d40a	02e7d1ca	53571dae	7a3182a2
12a8dddec	fdaa335d	176f43e8	71fb46d4	38129022	ce949ad4	b84769ad	965bd862
82f3d055	66fb9767	15b80b4e	1d5b47a0	4cfde06f	c28ec4b8	57e8726e	647a78fc
99865d44	608bd593	6c200e03	39dc5ff6	5d0b00a3	ae63aff2	7e8bd632	70108c0c
bbd35049	2998df04	980cf42a	9b6df491	9e7edd53	06918548	58cb7e07	3b74ef2e
522ffffb1	d24708cc	1c7e27cd	a4eb215b	3cf1d2e2	19b47a38	424f7618	35856039
9d17dee7	27eb35e6	c9aff67b	36baf5b8	09c467cd	c18910b1	e11dbf7b	06cd1af8
7170c608	2d5e3354	d4de495a	64c6d006	bcc0c62c	3dd00db3	708f8f34	77d51b42
264f620f	24b8d2bf	15c1b79e	46a52564	f8d7e54e	3e378160	7895cda5	859c15a5
e6459788	c37bc75f	db07ba0c	0676a3ab	7f229b1e	31842e7b	24259fd7	f8bef472
835ffcb8	6df4c1f2	96f5b195	fd0af0fc	b0fe134c	e2506d3d	4f9b12ea	f215f225
a223736f	9fb4c428	25d04979	34c713f8	c4618187	ea7a6e98	7cd16efc	1436876c
f1544107	bedeee14	56e9af27	a04aa441	3cf7c899	92ecbae6	dd67016d	151682eb
a842eedf	fdb60b4	f1907b75	20e3030f	24d8c29e	e139673b	efa63fb8	71873054
b6f2cf3b	9f326442	cb15a4cc	b01a4504	f1e47d8d	844a1be5	bae7dfdc	42cbda70
cd7dae0a	57e85b7a	d53f5af6	20cf4d8c	cea4d428	79d130a4	3486ebfb	33d3cddc
77853b53	37effcb5	c5068778	e580b3e6	4e68b8f4	c5c8b37e	0d809ea2	398feb7c
132a4f94	43b7950e	2fee7d1c	223613bd	dd06caa2	37df932b	c4248289	acf3ebc3
5715f6b7	ef3478dd	f267616f	c148cbe4	9052815e	5e410fab	b48a2465	2eda7fa4
e87b40e4	e98ea084	5889e9e1	efd390fc	dd07d35b	db485694	38d7e5b2	57720101
730edebc	5b643113	94917e4f	503c2fba	646f1282	7523d24a	e0779695	f9c17a8f
7a5b2121	d187b896	29263a4d	ba510cdf	81f47c9f	ad1163ed	ea7b5965	1a00726e
11403092	00da6d77	4a0cdd61	ad1f4603	605bdfb0	9eedc364	22ebe6a8	cee7d28a
a0e736a0	5564a6b9	10853209	c7eb8f37	2de705ca	8951570f	df09822b	bd691a6c
aa12e4f2	87451c0f	e0f6a27a	3ada4819	4cf1764f	0d771c2b	67cdb156	350d8384
5938fa0f	42399ef3	36997b07	0e84093d	4aa93e61	8360d87b	1fa98b0c	1149382c
e97625a5	0614d1b7	0e25244b	0c768347	589e8d82	0d2059d1	a466bb1e	f8da0a82
04f19130	ba6e4ec0	99265164	1ee7230d	50b2ad80	eaee6801	8db2a283	ea8bf59e

Appendix B. Test Vectors

This appendix provides test vectors for the CAST-128 cipher described in this document.

B.1. Single Plaintext-Key-Ciphertext Sets

In order to ensure that the algorithm is implemented correctly, the following test vectors can be used for verification (values given in hexadecimal notation).

```

128-bit key      = 01 23 45 67 12 34 56 78 23 45 67 89 34 56 78 9A
    plaintext    = 01 23 45 67 89 AB CD EF
    ciphertext   = 23 8B 4F E5 84 7E 44 B2

80-bit  key      = 01 23 45 67 12 34 56 78 23 45
                  = 01 23 45 67 12 34 56 78 23 45 00 00 00 00 00 00
    plaintext    = 01 23 45 67 89 AB CD EF
    ciphertext   = EB 6A 71 1A 2C 02 27 1B

40-bit  key      = 01 23 45 67 12
                  = 01 23 45 67 12 00 00 00 00 00 00 00 00 00 00 00
    plaintext    = 01 23 45 67 89 AB CD EF
    ciphertext   = 7A C8 16 D1 6E 9B 30 2E

```

B.2. Full Maintenance Test

A maintenance test for CAST-128 has been defined to verify the correctness of implementations. It is defined in pseudo-code as follows, where *a* and *b* are 128-bit vectors, *aL* and *aR* are the leftmost and rightmost halves of *a*, *bL* and *bR* are the leftmost and rightmost halves of *b*, and *encrypt(d,k)* is the encryption in ECB mode of block *d* under key *k*.

```

Initial a = 01 23 45 67 12 34 56 78 23 45 67 89 34 56 78 9A (hex)
Initial b = 01 23 45 67 12 34 56 78 23 45 67 89 34 56 78 9A (hex)

```

```

do 1,000,000 times
{
    aL = encrypt(aL,b)
    aR = encrypt(aR,b)
    bL = encrypt(bL,a)
    bR = encrypt(bR,a)
}

```

```

Verify a == EE A9 D0 A2 49 FD 3B A6 B3 43 6F B8 9D 6D CA 92 (hex)
Verify b == B2 C9 5E B0 0C 31 AD 71 80 AC 05 B8 E8 3D 69 6E (hex)

```