INTERNATIONAL STANDARD

ISO/IEC 29192-3

First edition
2012-10-01

Information technology — Security techniques — Lightweight cryptography —

Part 3: Stream ciphers

Technologies de l'information — Techniques de sécurité — Cryptographie pour environnements contraints — Partie 3: Chiffrements à flot

Reference number
ISO/IEC 29192-3:2012(E)

ISO copyright office
Case postale 56 • CH-1211 Geneva 20
Tel. + 41 22 749 01 11
Fax + 41 22 749 09 47
E-mail copyright@iso.org
Web www.iso.org

Published in Switzerland

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

ISO/IEC 29192-3 was prepared by Joint Technical Committee ISO/IEC JTC 1, *Information technology*, Subcommittee SC 27, *IT Security techniques*.

ISO/IEC 29192 consists of the following parts, under the general title *Information technology — Security techniques — Lightweight cryptography*:

— *Part 1: General*

— *Part 2: Block ciphers*

— *Part 3: Stream ciphers*

— *Part 4: Mechanisms using asymmetric techniques*

Introduction

This part of ISO/IEC 29192 specifies keystream generators for lightweight stream ciphers tailored for implementation in constrained environments. ISO/IEC 29192-1 specifies the requirements for lightweight cryptography. A stream cipher is an encryption mechanism that uses a keystream generator to generate a keystream to encrypt a plaintext in bitwise or block-wise manner.

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of patents.

ISO and IEC take no position concerning the evidence, validity and scope of these patent rights.

The holders of these patent rights have assured ISO and IEC that they are willing to negotiate licences under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holders of these patent rights are registered with ISO and IEC. Information may be obtained from:

René-MichaelCordes/ErnstSchobesberger/M&C ConsultInvest&TradeGmbH
LogoDynamic Unit GmbH
Prinz Eugen Strasse 52/9,
A-1040 Vienna
Austria

Hitachi Ltd.
IP Licensing Department
Intellectual Property Group
Marunouchi Center Building
6-1, Marunouchi 1-chome,
Chiyoda-ku,
Tokyo, 100-8220
Japan

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO and IEC shall not be held responsible for identifying any or all such patent rights.

ISO (www.iso.org/patents) and IEC (http://patents.iec.ch) maintain on-line databases of patents relevant to their standards. Users are encouraged to consult the databases for the most up to date information concerning patents.

Information technology — Security techniques — Lightweight cryptography —

Part 3:
Stream ciphers

1 Scope

This part of ISO/IEC 29192 specifies two dedicated keystream generators for lightweight stream ciphers:

- Enocoro: a lightweight keystream generator with a key size of 80 or 128 bits;
- Trivium: a lightweight keystream generator with a key size of 80 bits.

2 Normative reference

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 29192-1, *Information technology — Security techniques — Lightweight cryptography — Part 1: General*

3 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 29192-1 and the following apply.

3.1
big-endian
method of storage of multi-byte numbers with the most significant bytes at the lowest memory addresses

[ISO/IEC 18033-4:2011]

3.2
ciphertext
data which has been transformed to hide its information content

[ISO/IEC 18033-1:2005]

3.3
decryption
reversal of a corresponding encipherment

[ISO/IEC 18033-1:2005]

3.4
encryption
(reversible) transformation of data by a cryptographic algorithm to produce ciphertext, i.e. to hide the information content of the data

[ISO/IEC 18033-1:2005]

3.5
initialization value
value used in defining the starting point of an encryption process

[ISO/IEC 18033-4:2011]

3.6
key
sequence of symbols that controls the operation of a cryptographic transformation (e.g. encipherment, decipherment)

[ISO/IEC 18033-1:2005]

3.7
keystream function
function that takes as input, the current state of the keystream generator and (optionally) part of the previously generated ciphertext, and gives as output the next part of the keystream

[ISO/IEC 18033-4:2011]

3.8
keystream generator
state-based process (i.e., a finite state machine) that takes as input, a key, an initialization vector, and if necessary the ciphertext, and gives as output a keystream (i.e., a sequence of bits or blocks of bits) of arbitrary length

[ISO/IEC 18033-4:2011]

3.9
next-state function
function that takes as input, the current state of the keystream generator and (optionally) part of the previously generated ciphertext, and gives as output a new state for the keystream generator

[ISO/IEC 18033-4:2011]

3.10
plaintext
unenciphered information

[ISO/IEC 18033-1:2005]

3.11
secret key
key used with symmetric cryptographic techniques by a specified set of entities

[ISO/IEC 18033-1:2005]

3.12
state
internal state of a keystream generator

4 Symbols and operational terms

0x	Prefix for hexadecimal values.
$0^{(n)}$	n-bit variable where 0 is assigned to every bit.
AND	Bitwise logical AND operation.
a_i	Variable forming part of the internal state of a keystream generator.
b_i	Variable forming part of the internal state of a keystream generator.
C_i	Ciphertext block.
$F[x]$	The polynomial ring over the finite field F.
$\mathrm{GF}(2^n)$	Finite field of 2^n elements.
Init	Function which generates the initial internal state of a keystream generator.
IV	Initialization vector.
K	Key.
Next	Next-state function of a keystream generator.
n	Block length.
OR	Bitwise logical OR operation.
Out	Output function combining keystream and plaintext in order to generate ciphertext.
P	Plaintext.
P_i	Plaintext block.
Strm	Keystream function of a keystream generator.
S_i	Internal state of a keystream generator.
Z	Keystream.
Z_i	Keystream block.
$\lceil x \rceil$	The smallest integer greater than or equal to the real number x.
$\neg x$	Bitwise complement operation.
$\bullet$	Polynomial multiplication.
$\|\|$	Bit concatenation.
$+_m$	Integer addition modulo 2^m.
$\oplus$	Bitwise XOR (eXclusive OR) operation.
$<<_n t$	t-bit left shift in an n-bit register.

$>>_n t$ *t*-bit right shift in an *n*-bit register.

$<<<_n t$ *t*-bit left circular rotation in an *n*-bit register.

$>>>_n t$ *t*-bit right circular rotation in an *n*-bit register.

$\otimes$ Multiplication operation for elements in the finite field GF(2^n).
NOTE An example of operation of multiplication of elements in the finite field GF(2^n) is given in Annex E.

5 General models for stream ciphers

5.1 General

This clause describes general models for stream ciphers [ISO/IEC 18033-4:2011].

5.2 Synchronous Keystream generators

A synchronous keystream generator is a finite-state machine. It is defined by:

1. An initialization function, *Init*, which takes as input a key K and an initialization vector IV, and outputs an initial state S_0 for the keystream generator. The initialization vector should be chosen so that no two messages are ever encrypted using the same key and the same IV.
2. A next-state function, *Next*, which takes as input the current state of the keystream generator S_i, and outputs the next state of the keystream generator S_{i+1}.
3. A keystream function, *Strm*, which takes as input a state of the keystream generator S_i, and outputs a keystream block Z_i.

When the synchronous keystream generator is first initialized, it will enter an initial state S_0 defined by

$S_0 = Init(IV, K)$.

On demand the synchronous keystream generator will for i=0,1,...:

1. Output a keystream block $Z_i = Strm(S_i, K)$.
2. Update the state of the machine $S_{i+1} = Next(S_i, K)$.

Therefore to define a synchronous keystream generator it is only necessary to specify the functions *Init, Next* and *Strm*, including the lengths and alphabets of the key, the initialization vector, the state, and the output block.

5.3 Output functions

5.3.1 General model of output function

This subclause specifies a stream cipher output function, i.e. a technique to be used in a stream cipher to combine a keystream with plaintext to derive ciphertext.

An output function for a synchronous or a self-synchronizing stream cipher is an invertible function *Out* that combines a plaintext block P_i, a keystream block Z_i to give a ciphertext block C_i ($i \geq 0$). A general model for a stream cipher output function is now defined.

Encryption of a plaintext block P_i by a keystream block Z_i is given by:

$C_i = Out(P_i, Z_i)$,

and decryption of a ciphertext block C_i by a keystream block Z_i is given by:

$P_i = Out^{-1}(C_i, Z_i)$.

The output function shall be such that, for any keystream block Z_i, and plaintext block P_i, we have

$P_i = Out^{-1}(Out(P_i, Z_i), Z_i)$.

5.3.2 Binary-additive output function

A binary-additive stream cipher is a stream cipher in which the keystream, plaintext, and ciphertext blocks are binary digits, and the operation to combine plaintext with keystream is bitwise XOR. Let n be the bit length of P_i. This function is specified by

$Out(P_i, Z_i) = P_i \oplus Z_i$.

The operation Out^{-1} is specified by

$Out^{-1}(C_i, Z_i) = C_i \oplus Z_i$.

6 Dedicated keystream generators

6.1 *Enocoro*-128v2 keystream generator

6.1.1 Introduction to *Enocoro*-128v2

Enocoro-128v2 is a keystream generator which uses a 128-bit secret key K, a 64-bit initialization vector IV, and a state variable S_i ($i \geq 0$) consisting of 34 bytes, and outputs a keystream block Z_i of one byte at every iteration of the function *Strm*.

NOTE This keystream generator was originally proposed in [5].

The state variable S_i is sub-divided into a 2-byte variable:

$a^{(i)} = (a_0^{(i)}, a_1^{(i)})$,

where $a_j^{(i)}$ is a byte (for $j = 0, 1$), and a 32-byte variable:

$b^{(i)} = (b_0^{(i)}, b_1^{(i)}, \ldots, b_{31}^{(i)})$,

where $b_j^{(i)}$ is a byte (for $j = 0, 1, \ldots, 31$).

The *Init* function, defined in detail in 6.1.2, takes as input the 128-bit key K and the 64-bit initializing vector IV, and produces the initial value of the state variable $S_0 = (a^{(0)}, b^{(0)})$.

The *Next* function, defined in detail in 6.1.3, takes as input the 34-byte state variable $S_i = (a^{(i)}, b^{(i)})$ and produces as output the next value of the state variable $S_{i+1} = (a^{(i+1)}, b^{(i+1)})$.

The *Strm* function, defined in detail in 6.1.4, takes as input the 34-byte state variable $S_i = (a^{(i)}, b^{(i)})$ and produces as output the keystream block Z_i.

Enocoro-128v2 uses operations over the finite field $GF(2^8)$. In the polynomial representation, $GF(2^8)$ is realized as $GF(2)[x] / \phi_{8432}(x)$, where $\phi_{8432}(x)$ is an irreducible polynomial of degree 8 defined over $GF(2)$. The *Enocoro*-128v2 keystream generator uses the following irreducible polynomial:

$\phi_{8432}(x) = x^8 + x^4 + x^3 + x^2 + 1$.

6.1.2 Initialization function *Init*

The initialization of *Enocoro*-128v2 is divided into six steps. During the initialization of *Enocoro*-128v2, the state is updated as sketched in Figure 1.

The initialization function *Init* is as follows:

Input: 128-bit key K, 64-bit initialization vector IV.

Output: Initial value of the state variable $S_0 = (a^{(0)}, b^{(0)})$.

a) Use the key K to set part of the state variable $b_j^{(-96)}$ as follows:

- Set $(K_0||K_1||...||K_{15}) = K$, where K_j is 8 bits for j=0,1,2,...,15.
- For j=0,1,2,...,15, set $b_j^{(-96)} = K_j$.

b) Use the initialization vector *IV* to set part of the state variable $b_j^{(-96)}$ as follows:

- Set $(I_0||I_1||...||I_7) = IV$, where I_j is 8 bits for j=0,1,2,...,7.
- For j=0,1,2,...,7, set $b_{j+16}^{(-96)} = I_j$.

c) Use the constants $C_0, C_1, ..., C_9$ to set part of the state variable $a_j^{(-96)}$ and $b_j^{(-96)}$ as follows:

- Set $b_{24}^{(-96)} = C_0$ = 0x66,
- Set $b_{25}^{(-96)} = C_1$ = 0xe9,
- Set $b_{26}^{(-96)} = C_2$ = 0x4b,
- Set $b_{27}^{(-96)} = C_3$ = 0xd4,
- Set $b_{28}^{(-96)} = C_4$ = 0xef,
- Set $b_{29}^{(-96)} = C_5$ = 0x8a,
- Set $b_{30}^{(-96)} = C_6$ = 0x2c,
- Set $b_{31}^{(-96)} = C_7$ = 0x3b,
- Set $a_0^{(-96)} = C_8$ = 0x88,
- Set $a_1^{(-96)} = C_9$ = 0x4c.

d) Set an 8-bit counter ctr = 1.

e) Perform the following steps for i=-96,-95,...,-1:

- $b_{31}^{(i)} = b_{31}^{(i)} \oplus$ ctr,
- ctr = 0x02$\otimes$ ctr,
- Set S_{i+1} =Next(S_i).

f) Output S_0.