ISO/IEC JTC 1/SC 27 N17507

SC 27 SD12

REPLACES: N16827

ISO/IEC JTC 1/SC 27

Information technology - Security techniques

Secretariat: DIN, Germany

DOC TYPE:	Standing Document
TITLE:	**ISO/IEC JTC 1/SC 27 Standing Document No. 12 (SD12) on the Assessment of Cryptographic Techniques and Key Lengths, 5th edition**
SOURCE:	SD12 Editors (G.Vidal, H. von Sommerfeld and B. Poletti)
DATE:	2017-07-28
PROJECT:	SC 27 SD12
STATUS:	Draft Revised SD12
PLEASE NOTE:	**This document is also freely accessible from the public SC 27 web site at: http://www.jtc1sc27.din.de/sbe/sd12 Downloads**
ACTION ID:	FYI
DUE DATE:	
DISTRIBUTION:	P-, L-, O-Members W. Fumy, SC 27 Chairman M. De Soete, SC 27 Vice Chair E. Humphreys, T. Chikazawa, M. Bañón, J. Amsenga, K. Rannenberg, WG Convenors
MEDIUM:	http://isotc.iso.org/livelink/livelink/open/jtc1sc27
NO. OF PAGES:	1 + 25

Secretariat ISO/IEC JTC 1/SC 27 -
DIN Deutsches Institut für Normung e. V., Am DIN-Platz, Burggrafenstr. 6, D-10787 (postal D-10772) Berlin, Germany
Telephone: + 49 30 2601-2652; Facsimile:+ 49 30 2601-1723; E-Mail: krystyna.passia@din.de;
HTTP://www.jtc1sc27.din.de/en

ISO/IEC JTC 1/SC 27 Standing Document No. 12 (SD12) on the Assessment of Cryptographic Techniques and Key Lengths 5th edition

1 Disclaimer

The intention of this standing document is to provide general references to cryptographic techniques and key length selection sources as well as to present general background information that is applicable to the use of ISO/IEC cryptography standards. The references may be useful for users of the standards, but do not necessarily reflect the approval or disapproval of certain key-lengths or cryptographic techniques standardized within ISO/IEC, and particularly ISO/IEC JTC 1/SC 27. Applicable cryptographic techniques and key lengths are reflected by the International Standards developed within ISO/IEC JTC 1/SC 27 and should be referred if clarification is required.

2 Block Ciphers

A block cipher is a symmetric encipherment system with the property that the encryption algorithm operates on a block of plaintext, i.e. a string of bits of a defined length, to yield a block of ciphertext. The block ciphers are massively used in protocols nowadays and are involved in symmetric cryptography, hash functions, MAC constructions, etc.

2.1 Security analysis of two-key and three-key Triple-DES

General

The following is an analysis of three-key Triple DES (i.e. TDEA keying option 1) and two-key Triple DES (i.e. TDEA keying option 2), as standardized in ISO/IEC 18033-3:2005 Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers (see 4.1.2 ‘TDEA keying options’).

The encryption technique commonly known as Triple DES (Data Encryption Standard) is referred to as the Triple Data Encryption Algorithm (TDEA) by ISO/IEC 18033-3:2005 in 4.1 ‘TDEA’ (see [7]). Triple DES is a symmetric block cipher that can process data blocks of 64 bits, using keys with a length of 128 (or 192) bits, of which 112 (or 168) bits can be chosen arbitrarily and the rest may be used for error detection.

The effective strength of three-key Triple DES is at most 112 bits. The remainder of this clause focuses on the effective strength of two-key Triple DES.

The security of two-key Triple-DES

It is well-known that two-key Triple DES can be attacked in ways more efficient than a simple exhaustive search through the entire key space; such an exhaustive search attack would require of the order of 2^{112} encryption operations, and would be infeasible with current technology (and for many

years to come). However, as described by van Oorschot and Wiener [14], if 2^m plaintext/ciphertext pairs are known, all computed using the same secret key K, then K can be determined using of the order of 2^m storage and 2^{120-m} encryption/decryption operations. This is faster than an exhaustive search, as long as at least 2^8 (=256) plaintext/ciphertext pairs are known. As an example, in the case where as many as 2^{32} plaintext/ciphertext pairs are known (i.e. m=32), the attack requires 'only' 2^{88} encryption/decryption operations. Since this is beyond practical reach (in 2017), it is debatable whether this has a major impact on the security of two-key Triple DES in practical applications.

However, in more recent research [24], it has been shown that the van Oorschot-Wiener attack will still work even if the plaintext/ciphertext pairs available to a cryptanalyst were generated using a multiplicity of keys. That is, modifying the statement in the previous paragraph, if 2^m plaintext/ciphertext pairs are known, possibly computed using more than one secret key, then one of the secret keys can be determined using of the order of 2^m storage and 2^{120-m} encryption/decryption operations. This means that some of the precautions previously proposed to ensure the security of two-key Triple DES in practice, no longer apply. In particular, it was previously recommended in SD12 that, depending on the required security level, the maximum number of plaintexts encrypted under a single key should be limited. Whilst updating keys remains desirable (as always) and can limit the impact of a single key compromise, the more recent research findings mean that in any environment in which two-key Triple DES is used, if it is feasible for a cryptanalyst to obtain 2^m plaintext/ciphertext pairs (possibly generated using multiple keys), then the security of the algorithm is upper bounded by 2^{120-m} encryption/decryption operations. For example, if it is feasible for a cryptanalyst to accumulate 240 plaintext/ciphertext pairs, perhaps over a long period of time and using multiple keys, then of the order of 2^{80} DES encryption/decryption operations will be sufficient to find one of the keys.

The security of three-key Triple DES [This part was treating two-key Triple DES. Which one is accurate? Two-key or three-key?]

The security level of three-key Triple DES is less than the number of bits in a key (112 bits) because of a 'meet-in-the-middle' attack on the Triple DES construction. An attacker with access to 2^{40} plaintext/ciphertext pairs, possibly generated using multiple keys, will be able to launch a search for one of the keys used with effort equivalent to an exhaustive key search for an 80-bit key, using the attack described in [24]. The implication of this is that the effective key-length of two-key Triple DES in specific applications can only be regarded as being at most 80 bits (instead of 112 bits).

For many practical applications, this degradability of the effective key-length is not necessarily a problem as access to 2^{40} plaintext/ciphertext pairs may be deemed unlikely. However, conservative

system security design suggests that this assumption needs to be very carefully analyzed in the context of system use[1] before continuing to employ this encryption technique.

2.2 AES

General

The following is an analysis of the AES (Advanced Encryption Standard) in reference to ISO/IEC 18033-3:2005 Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers subclause 5.1 “AES”.

The AES is a symmetric block cipher which processes data blocks of 128 bits using a key of 128, 192 or 256 bits. Therefore, the algorithms with these different sizes of key will be referred as AES-128, AES-192 and AES-256 respectively to the size of the key. According to [26], the versions with longer key size are slower by 20% and 40% respectively.

The effective strength of these three versions of the cipher is equal to the size of the key (128 bits for AES-128, etc.).

The security of the AES

The AES is the legitimate choice for current and future applications. It has been precisely invented to resist to the linear and differential cryptanalysis for which the DES does not.

The AES can be attacked by single-key recovery methods as impossible differential cryptanalysis and Square attacks. Impossible differential cryptanalysis, instead of tracking the propagation of differences through the computation of the algorithm, tracks differences that have probability 0 to happen at a certain stage of the algorithm.

The AES-192 and the AES-256 are vulnerable to related-key attacks (see part related-key attacks) [27][28]. The best attack for the AES-192 requires data complexity 2^{123} and time complexity 2^{176}. For the AES-256, the attack is more performant and requires data complexity $2^{99.5}$ and time complexity $2^{99.5}$. These attacks work because of the difference between the key size and the block size which has impact on the key schedule. That is also the reason why AES-128 is not vulnerable. Nevertheless, taking independent and random keys avoid these attacks.

As the effective strength of the algorithm is 128 bits (respectively 192 and 256), a brute force attack leads to 2^{128} (2^{192} and 2^{256}) computations. The bi-clique technique presented in [29] reduces the complexity of the brute force attack. Bogdanov et al. show that it needs $2^{126.18}$ time complexity and

[1] It is noted in [24] that in some cases only partial plaintexts need to be known to the attacker.

2^{88} data complexity to break AES-128, $2^{189.74}$ time complexity and 2^{80} data complexity to break AES-192 and $2^{254.42}$ time complexity and 2^{40} data complexity to break AES-256.

Those few attacks, they are mostly theoretical for now and demand a lot of computing power. Using the AES in current and future applications is recommended.

2.3 Camellia

General

The following is an analysis of Camellia in reference to ISO/IEC 18033-3:2005 Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers subclause 5.2 “Camellia”.

Camellia is a symmetric block cipher processing data blocks of 128 bits using a key of 128, 192 or 256 bits. The two versions with 192 and 256 bits are both 33% slower than the 128 bits version.

The effective strength of these three versions of the cipher is equal to the size of the key.

Security analysis of Camellia

Camellia is a block cipher safe to use. The security provided by Camellia and the techniques used to compute it are comparable to those of AES. Another similarity is that it does not exist, at the moment of writing, any attack which breaks significantly the effective strength of the cipher.

However, because of its algebraic proprieties, Camellia is theoretically vulnerable to algebraic attacks which demands a number of computations way too elevated for the current computation capacities. Thus these attacks have been proved ineffective. Hence, Camellia is approved to be used in modern and future applications.

3 Modes of operation

A mode of operation is an algorithm which encrypts/decrypts a message by using a symmetric-key block cipher in order to provide confidentiality or authentication. In the following, five modes are presented and each is a reference to ISO/IEC 10116:2006 *Information technology — Security techniques — Modes of operation for an n-bit block cipher clause 6 to 10*.

3.1 Electronic CodeBook (ECB) mode

ECB is the simplest mode available. The message is divided into blocks and each block is encrypted separately (as well for the decryption).

The problem with this method is that the same plaintext blocks are encrypted into the same ciphertext blocks. Thus, the possible repetitions are not hidden, which is a confidentiality issue.

As a consequence, this mode of operation is not recommended.

3.2 Cipher Block Chaining (CBC) mode

The CBC mode is the most used in practice. The plaintext is divided in n blocks and each plaintext block is **XOR**-ed with the previous ciphertext block and is encrypted.

For the very first block, as there is not a “previous” ciphertext, an IV (Initial Vector) is used. The latter must be independent and random for each plaintext in order to make the algorithm indistinguishable under chosen plaintext attack.

Before the beginning of encryption, CBC needs to use padding on its plaintext but several padding oracle attacks exist and weaken this mode.

3.3 Cipher FeedBack (CFB) mode

In CFB, the current ciphertext block is created with the XOR of the current plaintext block and the output of the encryption of the previous plaintext block.

CFB requires an independent and random IV but it does not need to be secret.

It transforms a block cipher into a self-synchronizing stream cipher. A ciphertext block, instead of depending of a plaintext block and a key (like ECB), depends on the previous ciphertext block, that is to say depends on the whole plaintext.

To proceed the decryption, the user only needs the encryption scheme of the block cipher. In other words, when the block cipher encryption and block cipher decryption is different (e.g. the AES), the block cipher decryption does not need to be implemented within the CFB algorithm.

Besides, this mode does not use padding, thus it is not vulnerable to oracle padding attacks.

However, when an error occurs, it is propagated through the rest of the plaintext but the user only lost a certain part of the plaintext because of its self-synchronizing characteristic.

3.4 Output FeedBack (OFB) mode

OFB transforms a block cipher into a synchronous stream cipher. The current ciphertext block is created with the XOR of the current plaintext block and a keystream block. These keystream blocks are created by encrypting an input block for each XOR. At the first block, the input is an IV and for the next ones it is the previous output of the encryption.

This mode requires a unique IV (must be a nonce) for each plaintext, if not the security is compromised.

Moreover, the security may also be compromised if any of the input block (used to create keystream block in order to encrypt the message) is used for another message.

In OFB, there are no dependencies between the ciphertext block. Error correcting codes are used to fix a flip of one or more bits. Because of its no-dependency characteristic, this mode is used when the user cannot tolerate error propagation.

AS CFB, OFB does not use padding and only needs the encryption scheme of the block cipher for the algorithm.

3.5 Counter (CTR) mode

As the two last modes of operation, CTR transforms a block cipher into a stream cipher.

The current ciphertext block is created with the XOR of the current plaintext block and the keystream block. Each successive keystream block is created by encrypting an input block called counter block. Each block of the counter must be distinct with every each other counter block.

Moreover, if several plaintexts are encrypted under the same key (of the block cipher), all the counter blocks from each plaintext must be distinct too.

The counter can be any function which outputs a sequence. This sequence must not repeat for a long time. The increment-by-one function is mostly chosen and is the simplest. Each block of the counter can be concatenated or added or XOR-ed with a random nonce before being encrypted by the block cipher. If the nonce is not random, it should only be concatenated in order to prevent security issues.

As OFB, this mode does not have plaintext dependency, each ciphertext block does not depend on the previous one.

As CFB and OFB, CTR does not use padding and only needs the encryption scheme of the block cipher for the algorithm.

Moreover, CTR does not propagate error of transmission.

However, the counter must be synchronized between the sender and the receiver. If not, errors during the decryption occur.

4 Stream Ciphers

A stream cipher is a symmetric key algorithm which uses a keystream to encrypt a plaintext in a bitwise or block-wise manner. A keystream is a sequence of bits or blocks of bits.

Two types of stream cipher exist: the self-synchronizing stream cipher (e.g. see the CFB mode previously) and the synchronous stream cipher (e.g. see the OFB mode previously).

4.1 SNOW 2.0

General

The following is an analysis of SNOW 2.0 in reference to ISO/IEC 18033-4:2011 Information technology – Security techniques – Encryption algorithms – Part 4: Stream ciphers subclause 8.2 “SNOW 2.0 key stream generator”.

It is a synchronous stream cipher with two variants: a 128-bits and 256-bits key and a 32-bits IV. It is used within the 3GPP Confidentiality and Integrity Algorithms UEA2 and UIA2.

Security analysis of SNOW 2.0

There is a distinguishing attack on SNOW 2.0 presented in [30] theoretically feasible but needs 2^{174} bits of keystream and 2^{174} time complexity.

In [31], it is shown that a modified version of SNOW 2.0 is vulnerable to algebraic attacks which requires about 2^{50} time complexity and no more than 1000 outputs.

Kircanski presented in his thesis [32] a sliding attack recovering related-key pair sets for SNOW 2.0. A related-key attack can be mounted with these results on SNOW 2.0, the 256 bits version. It raises questions about the validity of the security protocols SNOW 2.0.

4.2 Rabbit

General

The following is an analysis of Rabbit in reference to ISO/IEC 18033-4:2011 Information technology – Security techniques – Encryption algorithms – Part 4: Stream ciphers subclause 8.3 “Rabbit key stream generator”. It was submitted to the eSTREAM competition and included in the final eSTREAM portofolio for the hardware profile.

It is a synchronous stream cipher using a 128-bits key and a 64-bits IV (public) which has been designed for high performance in software implementations. This cipher also turns out to have high performance in hardware implementations.

Security analysis of Rabbit

Rabbit is supposed to provide a 128-bits security strength. But, due to generic TMD trade-off attacks [33], if an attack targets a lot of keys at once and does not put any interest at which one he breaks, then with the small size of the IV it results that the security strength of the algorithm decreases to 96 bits.

Aumasson presented in [34] the existence of a bias in the output of Rabbit that leads to a distinguisher which requires about 2^{147} samples of keystream of 128 bits. Nevertheless the cost of this attack is much higher than an exhaustive search (2^{128} operations).

4.3 Trivium

General

The following is an analysis of Trivium in reference to ISO/IEC 29192-3:2012 Information technology – Security techniques – Lightweight cryptography — Part 3: Stream ciphers subclause 6.3 “Trivium keystream generator”. It was submitted to the eSTREAM competition and included in the final eSTREAM portofolio for the hardware profile.

It is a synchronous stream cipher using an 80-bits key and an 80-bits IV which has been designed for high performance in hardware implementation.

Security analysis of Trivium

In [35], Aumasson et al. presented cube testers which gives a distinguishing attack on reduced Trivium (790 out of 1152).

In [36], Maximov and Biryukov presented an attack recovering the internal state (and thus the secret key) with time complexity around $2^{83.5}$. The authors showed also that increasing the size of the key does not lead to an increase of the security strength of the algorithm. Modifying Trivium to provide 128-bits security strength or more is an open problem.

5 Hash functions

The following is in reference to ISO/IEC 10118-1:2016 Information technology — Security techniques — Hash-functions.

A hash function is a function which maps strings of bits of variable (but usually upper bounded) length to fixed-length strings of bits, satisfying the following two properties:

- for a given output, it is computationally infeasible to find an input which maps to this output;
- for a given input, it is computationally infeasible to find a second input which maps to the same output.

5.1 SHA-2

General

The following is an analysis of SHA-2, which is a set of six hash functions SHA-224, SHA-256, SHA-384, SHA-512, SHA-512/224 and SHA-512/256, in reference to ISO/IEC 10118-3:2004: Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash-functions.

The hashes are respectively of length 224, 256, 384 and 512 bits. For SHA-224 and SHA-256, the max plaintext size is $2^{64} - 1$. For the others, the max plaintext size is $2^{128} - 1$.

SHA-256 and SHA-512 are computed with 32 and 64 words respectively. SHA-224 and SHA 384 are truncated versions of the first two with different initial vectors. SHA-512/224 and SHA-512/256 are also truncation of the first two but the initial values are generated differently.

Security analysis of SHA-2

Several pre-image attacks have been presented on variant of SHA-256 and SHA-512.

Khovratovich et al. [37] reported a biclique pre-image attack breaking a reduced version of SHA-256 (52 rounds out of 64) with time complexity 2^{255} and a reduced version of SHA512 (57 rounds out of 80) with time complexity 2^{511}.

Lamberger and Mendel [38] presented a differential pseudo-collision attack which breaks 46 rounds out of 64 of SHA-256.

The full version of the family functions SHA-2 have not been broken.

5.2 WHIRLPOOL

General

The following is an analysis of WHIRLPOOL, a hash function in reference to ISO/IEC 10118-3:2004 Information technology -- Security techniques -- Hash-functions -- Part 3: Dedicated hash-functions subclause 13 “Dedicated Hash-Function 7 (WHIRLPOOL).

WHIRLPOOL takes as input plaintext at most $2^{256} - 1$ bits and returns a ciphertext of 512 bits length. Its construction is based on the Advanced Encryption Standard (AES) block cipher.

Security analysis of WHIRLPOOL

In [39], the authors presented three attacks on reduced WHIRLPOOL function. First a collision attack on 4.5 rounds of WHIRLPOOL (it has 10 rounds in the full version) with 2^{120} time complexity and 2^{16} data complexity. The second attack is a semi-free-start collision attack on 5.5 rounds with 2^{120} time complexity and 2^{16} data complexity. Finally a semi-free-start near-collision attack on 7.5 rounds with 2^{128} time complexity and 2^{16} data complexity.

Others collisions attacks have been presented in [40] on reduced Whirlpool (e.g. with $2^{120}/2^{64}$ and $2^{112}/2^{64}$ time complexity/data complexity).

In [41], Sasuki presented a Second Preimage attack on 5 rounds of WHIRLPOOL with 2^{504} time complexity and 2^{8} data complexity.

6 Asymmetric Cryptography

This part is in reference to the ISO/IEC 18033-2:2006 Information technology — Security techniques — Encryption algorithms — Part 2: Asymmetric ciphers.

An asymmetric cryptographic technique is a cryptographic technique that uses two related transformations, a public transformation (defined by the public key) and a private transformation /defined by the private key). The two transformations have the property that, given the public transformation, it is computationally infeasible to derive the private transformation.

The most widespread uses of asymmetric cryptography are on one hand as an asymmetric cipher, with the public transformation used as the encryption and the private transformation as the decryption. The cipher allows a sender to use a recipient's public key to transmit an encryption of a message to the receiver who can use his secret key to decrypt the given ciphertext, thereby obtaining the original message. On the other hand, as a digital signature, to provide sender authentication, integrity authentication and support for non-repudiation.

The combination of the enciphering and the digital signature constitutes the Enveloped Public Key Encryption.

6.1 The factoring and RSA problem

The factoring problem

The factoring problem is the underlying hard problem for all schemes within the RSA family.

The factoring problem stipulates that it is computationally feasible to multiply two distinct large prime numbers but it is computationally not feasible to find those two prime numbers only from their product.

With the RSA Factoring Challenge, the cutting edge in integer factorization was tracked. The last factorized integers were RSA-576 in 2003, RSA-640 in 2005 an RSA-768 in 2009 using the Number Field Sieve algorithm [42] which is the most efficient factorization algorithm for large integers (> 100 decimal digits). Thus it is recommended to use a key of 1024 bits at least for legacy-use and a key of 3072 bits at least for future applications.

The product of the two distinct large prime numbers is usually denoted N=p·q. To assure the difficulty to factorize them, p and q should be of the same bit-length but not too close together. Indeed, if e.g. length (p) << length (q), it is easier to factorize their product by using the ECM method presented by Lenstra in 1987 [43]. Moreover, according to [44], if $|p - q| < N^{1/4}$, N can be factoring.

In a nutshell, p and q should be selected randomly and of length the half of N in order to ensure that it is difficult to factorize.

In the quantum cryptography, Shor's algorithm [45] resolves the factoring problem in polynomial time in the input size, e.g. the number of digits of the number to be factorize. This breakthrough would annihilate RSA based algorithms but it needs the development of efficient quantum computers, which is still in its research period for now.

The RSA problem

The RSA problem stipulates that, giving the composite modulus N = p.q, e the public exponent prime to (p-1)(q-1) and $c = m^e \ mod \ N$, it is hard to find m.

Actually, the ciphers based on the factoring problem are more based on the RSA problem, which is a priori as hard as the factoring problem.

Thus e should be large enough to avoid the solution of this problem, e.g. by using Coppersmith's lattice based techniques (e.g. [46]). Currently, it is recommended to take e ≥ 65537 [47] for encryption schemes which provides the best quality (security level)/speed ratio. For signature schemes, users usually take e = 3 to be as fast as possible despite the lack of security. For future applications, the minimum to use is e = 65537.

The developer should be careful also with the value of d, the private exponent and inverse of e. Indeed, a d too small (to have faster calculations) induces security issues. Lattice attacks can be applied when d is too small (e.g. [48]). Selecting e first and then finding d according to e, as standard practice, should lead to a d large enough.

7 Recommended algorithms and key size versus time

Here are the definitions used throughout the recommendations:

Approved: means that the algorithm or the key size is recommended to be used to apply cryptographic protection on data or to process cryptographic protection on encrypted data. Moreover, often their security has been proved.

Acceptable: means that the algorithm or the key size can be used to apply cryptographic protection on data or to process cryptographic protection on encrypted data but better alternatives exist.

Legacy-use: means that the algorithm or the key size could be used only to process already-encrypted data.

Disallowed: means that the algorithm or the key size shall not be used to apply cryptographic protection on data.

These recommendations have to be considered having consciousness that a breakthrough can happened (but is improbable) and discredits the recommendations. Moreover, even if the key size is

recommended, nothing guarantees that a cipher using this key size in a few years would still be secure even if it does not exist a computation power able to brute force this key. Thus, for the recommendations of 10 and 15 years, because of the uncertainty of the predictions, they can be qualified as speculative.

7.1 Block ciphers vs time

Here are the recommendations of the use of the block ciphers presented in the subclause 2 over the future time. Security strength is added to compare the block ciphers.

The effective strength of each cipher has been mentioned and can be compared to the key size as input.

<table>
<tr><th>Algorithm (key length)</th><th>Effective Strength</th><th>Mode</th><th>2018</th><th>2022</th><th>2027</th><th>2032</th></tr>
<tr><td rowspan="2">Two-Key TDES
(128 bits)</td><td rowspan="2">80</td><td>Encryption</td><td colspan="4">Disallowed</td></tr>
<tr><td>Decryption</td><td colspan="4">Legacy-use</td></tr>
<tr><td rowspan="2">Three-Key TDES
(192 bits)</td><td rowspan="2">112</td><td>Encryption</td><td colspan="3" rowspan="2">Acceptable</td><td>Dis-allowed</td></tr>
<tr><td>Decryption</td><td>Legacy-use</td></tr>
<tr><td>Camellia
(128, 192 and 256 bits)</td><td>128/192/256</td><td>En/Decryption</td><td colspan="4">Approved</td></tr>
<tr><td>AES
(128, 192 and 256 bits)</td><td>128/192/256</td><td>En/Decryption</td><td colspan="4">Approved</td></tr>
</table>

7.2 Stream ciphers versus time

Here are the recommendations of the use of the stream ciphers presented in the subclause 4 over the future time. Characteristics as security strength and IV length are added to compare the stream ciphers.

Algorithm (key length)	Effective Strength	IV length	2018	2022	2027	2032
SNOW 2.0 (128/256 bits)	128/256	32	Approved			

Rabbit (128 bits)	128	128	Legacy-use	Disallowed
Trivium (80 bits)	80	80	Legacy-use	Disallowed

7.3 Hash functions versus time

Here are the recommendations of the use of the hash functions presented in the subclause 5 over the future time. Characteristics as output length, security strength and max plaintext length are added to compare the hash functions.

<table>
<tr><th>Hash family</th><th>Hash function</th><th>Output length</th><th>Security strength</th><th>Max plaintext length</th><th>2018</th><th>2022</th><th>2027</th><th>2032</th></tr>
<tr><td rowspan="6">SHA-2</td><td>SHA-224</td><td>224</td><td>112</td><td rowspan="2">$2^{64}-1$</td><td colspan="3" rowspan="2">Acceptable</td><td>Dis-allowed</td></tr>
<tr><td>SHA-256</td><td>256</td><td>128</td><td colspan="4" rowspan="5">Approved</td></tr>
<tr><td>SHA-384</td><td>384</td><td>192</td><td rowspan="4">$2^{128}-1$</td></tr>
<tr><td>SHA-512</td><td>512</td><td>256</td></tr>
<tr><td>SHA-512/224</td><td>224</td><td>112</td></tr>
<tr><td>SHA-512/256</td><td>256</td><td>128</td></tr>
<tr><td>-</td><td>WHIRLPOOL</td><td>512</td><td>256</td><td>$2^{256}-1$</td><td colspan="4">Approved</td></tr>
<tr><td>-</td><td>SHA-1</td><td>160</td><td>< 80</td><td>$2^{64}-1$</td><td colspan="4">Disapproved</td></tr>
</table>

7.4 Key size versus time

The following table shows the minimum key size that should be used according the cryptographic algorithm and the type of recommendation the developer want for his application or future application.

More precisely, in that context:

Approved means that he should use this key size at least if he want his application to work on plain data for a short and large duration (> 3 years).

Acceptable means that he should use this key size if he want his application to work on plain data for a short duration (≤ 2-3 years).

Legacy-use means that he can use this key size if his application works on already encrypted data.

Disallowed means he should not use this size to have proper security within his application.

Recommendation type	Symmetric key	Factoring Modulus	Hash
Disallowed (current applications)	< 112	≤ 1024	< 224
Legacy-use (current applications)	= 112	= 2048	= 224
Acceptable (current applications)	= 128	= 3072	= 256
Approved (current and future applications)	≥ 256	≥ 3072	≥ 512

Here the focus is on the recommendations of the minimum effective strength which should be provided by the cryptographic algorithms (symmetric or asymmetric cryptography). The effective strength s corresponds to the number of operations 2^s that an attack should take to break the security of the cryptographic algorithm. It is not necessarily equal to the size of the key used within the cryptographic algorithm.

<table>
<tr><th>Effective Strength</th><th>Mode</th><th>2018</th><th>2022</th><th>2027</th><th>2032</th></tr>
<tr><td rowspan="2">< 112
(e.g. 80 for two-key TDES)</td><td>Encryption</td><td colspan="4">Disallowed</td></tr>
<tr><td>Decryption</td><td colspan="4">Legacy-use</td></tr>
<tr><td rowspan="2">112</td><td>Encryption</td><td colspan="3" rowspan="2">Acceptable</td><td>Disallowed</td></tr>
<tr><td>Decryption</td><td>Legacy-use</td></tr>
<tr><td>128</td><td>En/Decryption</td><td colspan="4">Approved</td></tr>
<tr><td>192</td><td>En/Decryption</td><td colspan="4">Approved</td></tr>
<tr><td>256</td><td>En/Decryption</td><td colspan="4">Approved</td></tr>
</table>

8 General references to cryptographic algorithm and key length selection

The selection of cryptographic algorithms and/or key-lengths is not an exact science. For this reason one may find conflicting recommendations for cryptographic algorithm and key length selection. Cryptographic algorithm and key length selection amongst other things depend on:

- The specific area of application (for example government vs commercial applications),
- The period of time the protected data shall remain secure,
- The amount of data that is to be encrypted without rekeying,
- Cost-relevant aspects such as the period of time a specific piece of equipment is expected to be used before a planned equipment upgrade can take place,
- Additional security margin requirements.

The security analyst should determine the security requirements to determine an appropriate key length and an appropriate cryptographic algorithm that fit the cost-relevant requirements and meet the margins of derived security parameters. The documents and web-sites listed in the bibliography contain background information which can aid in the cryptographic algorithm and key length selection process (see 5 below).

References [1] and [2] are of a general theoretical nature, while reference [3], [8], [9] and [10] contain recommendations which are aimed at specific applications or sectors. Reference [4] contains more references itself to key length selection criteria and also provide implementations of calculations of the various references above (and more) which may aid in key length selection. Reference [8] is a technical report specifically aimed at the financial services sector and takes interoperability into account in its recommendations.

9 Block length selection

A summary of the guidance provided here is this: for applications where large amounts of data might be encrypted using a single key, block ciphers with 128-bit blocks (or larger) should be always used. In addition, where 128-bit block ciphers can be used, they should be used. For highly constrained applications where only small amounts of data can be encrypted using a single key, and where a 128-bit block cipher poses a cost or performance barrier to providing security, smaller block sizes may be appropriate.

This guidance is discussed in more detail in the remainder of this clause.

In the standard modes of operation, like cipher-block chaining mode (CBC), cipher feedback mode (CFB), counter mode (CTR), etc., a block cipher with an n-bit block begins to leak plaintext information as the amount of data encrypted using a single key approaches $2^{n/2}$ blocks. See, for example, [23].

For a block cipher with a 128-bit block (like AES, CLEFIA, LEA, SIMON-128 and SPECK-128), this will typically not be an issue, because for most applications the number of blocks encrypted with a single key will stay well below 2^{64}. Underlying these data leakage issues is the “birthday problem”: a collection of $2^{n/2-k}$ randomly chosen n-bit values, for k ≥ 0, will exhibit a collision —i.e., a repeated value— with probability about 2^{-2k-1} [NOTE: In fact 2^{-2k-1} is an upper bound on the collision probability]. $2^{n/2}$ is referred to as the “birthday bound,” and given this amount of data, a collision will happen with probability about $1 - e^{-1/2} \approx 0.39$, which is much higher than might naively be guessed. Attacks on modes typically exploit the fact that information can be leaked if a block is repeated (this is the case for CBC mode), or that with enough data it’s possible to distinguish a permutation, which has distinct outputs given distinct inputs, from a function, which can have the same output for two different inputs. (This is an issue for CTR mode.) For these attacks, the advantage of an adversary who makes at most $2^{n/2-k}$ queries to the block cipher is bounded above by the collision probability, which is ≤ 2^{-2k-1}. For any value of n, for the standard modes (excluding ECB mode, where collisions in inputs can easily be constructed and detected) if the amount of data encrypted using a single key stays well below $2^{n/2}$ blocks —i.e., if k is not too small— then these attacks are not a concern.

McGrew [22] presented attacks against n-bit block ciphers in CBC, CFB, and CTR modes that can recover an unknown plaintext values when the birthday bound is not respected. The collision-based attacks against CBC and CFB are straightforward and relatively inexpensive to carry out against 64-bit block ciphers; attacks against CTR are more involved, but are still feasible.

Additionally, according to [23] the birthday attack for the 64-bit block cipher requires about 800GB data and about 20-40 hours. While the attack for 64-bit block ciphers requires still huge data and long time, similar attacks for the 32- or 48-bit block cipher require only a few MB data with a few seconds or a few GB data with a few minutes, respectively.

Consequently, in order to prevent birthday attack in real world, block ciphers which block size is less than 64 bits are not recommended.

It is important to note that the idea that there are no security issues until the number of blocks encrypted using a single key reaches $2^{n/2}$ is incorrect. Rather, the security degrades as the number of blocks approaches this number. Thus, the number of blocks encrypted using a single key for an n-bit block cipher should be kept well under $2^{n/2}$. This is discussed further below.

[NOTE: the attacks on modes discussed here are not *key recovery* attacks; rather they expose some new plaintext values, given that an attacker has access to something approaching $2^{n/2}$ matched plaintext/ciphertext pairs. This of course is a serious issue in many contexts, as each unseen plaintext

value should be secret, but is not necessarily as serious as a key-recovery attack, which exposes every plaintext value.] **[Is this paragraph accurate?]**

For block sizes n with n smaller than 128, it is important in practice to ensure that there are limits on the amount of data encrypted using a single key. ISO/IEC 29192-2 includes PRESENT, which is a 64-bit block cipher. For n = 64, $2^{n/2}$ is just 2^{32}, which corresponds to 32 gigabytes of data. This means that the key for a 64-bit block cipher operating in a standard mode should be changed well before the birthday bound, i.e., well before 2^{32} blocks are encrypted. Ensuring that the key is changed after 2^{32-k} blocks are encrypted means that collisions occur, along with the corresponding potential compromise to security, with probability about 2^{-2k-1}. The user's assessment of the risk associated with a particular application should dictate how large a value of k is required.

64-bit block ciphers (in standard modes) are only appropriate for applications where small amounts of data will be encrypted using a single key, and where 128-bit block ciphers are not viable. They should never be used (in standard modes) for applications where the amount of data available to an adversary cannot be tightly controlled.

An example of an appropriate use for a 64-bit block cipher is RFID item tracking for items of low to moderate value, where each item has a unique key, and where any particular tag is expected to be queried only a handful of times over its lifetime. Here an adversary would have little incentive to carry out an attack whose cost would exceed the value of the item. Another appropriate use would be authentication of users by a central server, where queries in a challenge-response scheme could be controlled and not repeated, thus allowing more than $2^{n/2}$ encryptions.

Inappropriate uses would include, for example, file encryption on a desktop machine, TLS applications, etc. **[Expand on this?]**

96-bit versions of Simon and Speck are included in ISO/IEC 29192-2/AMD1. For these block sizes, birthday attacks can also be of concern, although the issue is not as serious as it is for 64-bit blocks ciphers. The birthday bound here is $2^{n/2} = 2^{48}$ which is 3 petabytes of data, and so it would be important for these block ciphers to keep the number of blocks encrypted using a single key well below this amount (i.e., make the limit 2^{48-k} for appropriate k, as discussed above). The range of appropriate use cases for 96-bit block ciphers broadens over those for 64-bit block ciphers, but for any application involving large amounts of data, one should use a 128-bit block cipher.

Encryption modes offering beyond-the-birthday-bound security can be used for block ciphers with 64- or 96-bit blocks to extend the amount of data which can be encrypted using a single key. See [25], for example, for modes requiring two calls to the encryption function, which effectively turn a block cipher into a *tweakable block cipher*, and provide security for up to the full 2^n encryptions. (This assumes the

cipher can be modeled as an *ideal cipher*, i.e., that the permutations associated to the various keys behave like independent random permutations. This is a heuristic that may be appropriate for the block ciphers included in or proposed for ISO/IEC 29192-2 (PRESENT, CLEFIA, SIMON, SPECK, LEA), but it not appropriate for some block ciphers, such as those obtained by the Even-Mansour construction.)

Finally, it is important to remark that in the case that a message authentication code (MAC) is based on a symmetric key block cipher, as seen discussions in NIST SP-800 38B, the default recommendation is to limit the key to no more than 2^{48} messages when the block size is 128 bits and 2^{21} messages when the block size is 64 bits in order to satisfy that the collision probability is respectively less than one in a billion and less than one in a million.

10 Related key attacks

10.1 General

Related-key attacks against a block-cipher rely on the following assumption: it is possible for an attacker to encrypt or decrypt messages under several different keys whose values are initially unknown, but where some mathematical relationship connecting the keys is known to the attacker.

This is a strong assumption which is not relevant for a large number of practical applications such as encryption or MAC using a long-term key. However, for "ad-hoc" applications which make use of a block-cipher known to be vulnerable to one or several related-key attacks, it is strongly recommended to assess the impact on the security. Indeed, it should not be possible for an attacker to encrypt/decrypt messages under several different keys by controlling the mathematical relationship connecting the keys in such a way that the constraints imposed by the attack (e.g. the number of related-keys required to mount the attack, the mathematical relationship connecting the keys required to mount the attack, etc.) are realistic.

As an illustration, it is not recommended to use a block cipher *A* which is known to be vulnerable to one or several related-key attacks as a cryptographic primitive in an ad-hoc authentication protocol such that the encryption key of the block cipher *A* is equal to *K* XOR *R* where *K* is a long-term secret key and *R* is a random value under the control of an eventual attacker.

The original ideas of related key attacks were introduced by Biham [17] and Knudsen [18], and many more articles followed.

10.2 Example

As an example the following trivial related key attack is applicable to all block ciphers.

A block cipher encrypts one plaintext under an unknown key *K*. The attacker can modify the key *K* to form key K_i by setting K_i equal to the logical AND between *K* and the bit pattern containing all ones

except in bit position *i*. This operation must be performed by the system (or in a more formal model, by an Oracle).

The same plaintext is encrypted again, and if the corresponding ciphertext differs from the sample ciphertext obtained from key *K,* the attackers knows that bit position *i* of *K* is a one, otherwise bit position *i* is a zero. By repeating the steps for each of the bits of *K,* the entire key can be recovered by the attacker. If *K* contains *n*-bits, the attacker requires *n* masks (or calls to the Oracle), and *n*+1 encryption calls.

The attack described above is described in terms of block ciphers, but is clearly applicable to more than just block ciphers. In many practical applications one does not expect such modifications to the encryption key to be allowable by the system. The described attack is the most basic form of related key attack, but many more exist that also rely on the structure of the particular block cipher.

11 Possible defects in International Standards

11.1 General

This clause deals with perceived defects that came to the attention of ISO/IEC JTC 1/SC 27 on its cryptography standards and possible ways to deal with these defects should there be concern for their continued use.

International Standards may contain cryptographic techniques for which, after publication, concerns are raised as to possible defects that may exist in the mechanisms. In some cases, these perceived defects are in fact not defects, but rather concerns expressed by the community (such as constants used in the mechanism which were generated in an unknown way). ISO/IEC JTC 1/SC 27 usually initiates a study period to which its liaison organizations, experts and National Bodies contribute. The outcome of such a study period can then either

1. Confirm the defect.
2. Prove the defect to be invalid.
3. Neither prove nor confirm the defect, but propose mitigation techniques.

In the case of 1. and 3. a Technical Corrigendum to the respective International Standard will be published. In all the cases further information can be made available in this document if appropriate.

11.2 MASH-1 hash function

ISO/IEC 10118-4:1998 *Information technology – Security techniques – Hash functions using an n-bit block cipher* contains a mechanism called MASH-1. To the knowledge of ISO/IEC JTC 1/SC 27 experts

a paper was published at the pre-proceedings of CTCrypt 2013 (also available on ePrint [16]). According to the paper it is possible to select weak moduli for use in MASH-1 which may lead to a weakening of the collision resistance of MASH-1. It is however possible to choose the moduli carefully so that this does not happen. A Technical Corrigendum is currently being produced by ISO/IEC JTC 1/SC 27 which will contain more detail on how to avoid choosing weak moduli.
Alternatively, users can study the original paper (see [16]) on how to avoid generating weak moduli.

11.3 Dual elliptic curve deterministic random bit generator (Dual_EC_DRBG)

ISO/IEC 18031:2011 *Information technology – Security techniques – Random bit generation* [6] contains a number of random bit generator mechanisms, one of which is known as the Dual_EC_DRBG. The same mechanism with identical application specific constants is specified in

U.S. National Institute of Standards and Technology (NIST) Special Publication 800-90A, which was a major contribution in establishing the International Standard.

Recent community commentary has called into question the trustworthiness of this mechanism. In particular these comments relate to the default elliptic curve points i.e. the default application specific constants that are provided in Annex D of this International Standard.

To the knowledge of ISO/IEC JTC 1/SC 27 experts no confirmation has surfaced to date that the default application specific constants actually lead to a compromise of the mechanism. However, there is sufficient evidence of a security issue. NIST has released a supplemental bulletin [12] giving advice about use of the mechanism and further U.S. federal government recommendations. In the outcome of the SC 27 Korea 2013 meeting ISO/IEC JTC 1/SC 27 has initiated a study period to carefully review the security issues and possible ramifications for the International Standard. This process runs in parallel to the NIST public comment period on the same mechanism and includes all national bodies and their experts subscribing that are members of ISO/IEC JTC 1/SC 27. The outcome of this study period may lead to a future revision of the International Standard.

ISO/IEC JTC 1/SC 27 is aware that Dual_EC_DRBG is included in many cryptographic libraries. Implementers and users of this mechanism should conduct a risk analysis of their security products, services and/or systems and may decide to take one of the following actions:

1. Continue using Dual_EC_DRBG if the perceived risk does not impact the security of their products, services and/or systems.

2. Generate their own application specific constants for application in Dual_EC_DRBG, and continue to use this random bit generator mechanism with the new constants.

3. Stop using Dual_EC_DRBG and replace it with another pseudo random bit generator from ISO/IEC 18031 *Information technology – Security techniques – Random bit generation* [6].

However, the second option (i.e. generate new specific constants) would require the appropriate level of technical expertise and some specific knowledge of Elliptic Curve Cryptography, and is only possible if the application is designed for allowing new constants to be specified.

ISO/IEC JTC 1/SC 27 expert discussions on [6], [20] and [21] during the SC 27 Korea 2013 meeting perceived, there is sufficient evidence of a security issue. The security of the Dual_EC_DRBG is based entirely on the property of two specific Dual_EC_DRBG system parameters (i.e. the a.m. application specific constants which are points on a given elliptic curve) to be chosen independently and random. As long as there is no confidence that the two parameters are actually chosen in this way, the Dual_EC_DRBG should be considered 'compromised'.

To the knowledge of the experts there is no proof positive that the default application specific constants given in [6] meet the requirements.

In the outcome of the SC 27 Hong Kong 2014 meeting the Dual_EC_DRBG mechanism (and associated material) will be removed from the text of ISO/IEC 18031, and a corrigendum will be prepared to achieve this objective. Further information the Dual_EC_DRBG mechanism will be made available in a new edition of SD 12 in parallel to publication of the corrigendum if appropriate.

11.4 Multivariate quadratic deterministic random bit generator (MQ_DRBG)

Another mechanism which is specified in ISO/IEC 18031:2011 *Information technology – Security techniques – Random bit generation* is multivariate quadratic deterministic random bit generator – MQ_DRBG. Particularly, this mechanism specifies restrictions on parameters of multivariate quadratic equations that are used for bit generation. However, the proof of security of the generator strongly relies on the randomness of the choice of the set multivariate quadratic equations. As it is shown in [19], the security level of the generator could be lower than specified in the International Standard, if the equations are not randomly selected. It is therefore required to randomly choose the equations in order to ensure the security of generator.

12 Development of quantum computers [title can be modified]

[As suggest in the comment RU2 for 54th ISO meeting in Hamilton, New Zealand, a clause on quantum computers should be treated. The editors of SC27 SD12 welcome the NBs to provide a text contribution regarding this topic.]

13 Bibliography

[1] Arjen K. Lenstra and Eric R. Verheul, Selecting Cryptographic Key Sizes, PKC2000: p. 446-465, 01/2000.

[2] European Network of Excellence in Cryptology, ECRYPT2 Yearly Report on Algorithms and Keysizes (2009-2010), D.SPA.13.

[3] H. Orman and P. Hoffman, Determining Strengths for Public Keys Used for Exchanging Symmetric Keys, RFC 3766, 04/2004.

[4] <http://www.keylength.com>.

[5] ISO/IEC 10118-4:1998, Information technology – Security techniques – Hash functions using an n-bit block cipher

[6] ISO/IEC 18031:2011 Information technology – Security techniques – Random bit generation

[7] ISO/IEC 18033-3:2005, Information technology – Security techniques – Encryption algorithms – Part 3: Block ciphers.

[8] ISO TR 14742, Financial services — Recommendations on cryptographic algorithms and their use, (ISO TC 68, Technical Report).

[9] National Security Agency (US), Fact Sheet Suite B Cryptography, 08/2009.

[10] NIST Special Publication 800-57, Recommendation for Key Management, Part 1: General (Revised), March 2007.

[11] NIST Special Publication 800-90A.

[12] NIST Special Bulletin <http://csrc.nist.gov/publications/nistbul/itlbul2013_09_supplemental.pdf>.

[13] NIST online document <http://csrc.nist.gov/groups/STM/cavp/documents/drbg/drbgval.html>.

[14] P. C. van Oorschot and M. J. Wiener, ‘A known-plaintext attack on two-key triple encryption’. In I. B. Damgård, ed., Proc. Eurocrypt ’90, Springer-Verlag LNCS 473 (1996) page 318-325.

[15] Robshaw, M (Ed), Fast Software Encryption 2006.

[16] V. G. Antipkin, “Smashing MASH-1”, *Math. Asp. of Crypt.*, **5**:2 (2014), 21–28

[17] Biham E, New types of cryptanalytic attacks using related keys." Journal of Cryptology, 4, Springer.

[18] Knudsen L.R., Cryptanalysis of LOKI91, Advances in Cryptography, Asiacrypt '92, LNCS 718, Springer-Verlag.

[19] Vladimir Drelikhov, Grigory Marshalko, Alexey Pokrovsky, On the security of MQ_DRBG, <eprint.iacr.org/2011/548>.

[20] Dan Shumow, Niels Ferguson, On the Possibility of a Back Door in the NIST SP800-90 Dual Ec Prng. CRYPTO Rump Session 2007. Microsoft, <http://rump2007.cr.yp.to/15-shumow.pdf>.

[21] United States Patent Application Publication, US 2007/0189527 A1, Aug. 16, 2007, “Elliptic Curve Random Number Generation”.

[22] David McGrew, Impossible plaintext cryptanalysis and probable-plaintext collision attacks of 64 bit block cipher modes" <eprint.iacr.org/2012/623>

[23] Karthikeyan Bhargavan and Gaëtan Leurent, On the Practical (In-)Security of 64-bit Block Ciphers: Collision Attacks on HTTP over TLS and OpenVPN, <eprint.iacr.org/2016/798>

[24] C. J. Mitchell, 'On the security of 2-key triple DES', *IEEE Transactions on Information Theory*, **62** (2016) 6260-6267.

[25] Lei Wang and Jian Guo and Guoyan Zhang and Jingyuan Zhao and Dawu Gu, How to build Fully Secure Tweakable Blockciphers from Classical Blockciphers, IACR eprint 2016/876.

[26] ENISA. Algorithms, Key Sizes and Parameters Report. 2013.

[27] Biryukov, Alex, et al. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. Henri Gilbert. EUROCRYPT 2010. s.l. : Springer, 2010, pp. 299-319.

[28] Biryukov, Alex and Khovratovich, Dmitry. Related-key cryptanalysis of the full AES-192 and AES-256. Mitsuru Matsui. ASIACRYPT 2009. Tokyo: Springer, 2009, pp. 1-18.

[29] Bogdanov, Andrey, Khovratovich, Dmitry and Rechberger, Christian. Biclique Cryptanalysis of the Full AES. 2011.

[30] Improved linear distinguishers for SNOW 2.0. Wallén, Johan and Nyberg, Kaisa. [ed.] Matthew J. B. Robshaw. s.l. : Springer, 2006, Lecture Notes in Computer Science, Vol. 4047, pp. 144-162.

[31] Billet, Olivier and Gilbert, Henri, Resistance of SNOW 2.0 Against Algebraic Attacks. [ed.] Alfred Menezes. s.l. : Springer, 2005, Lectures Notes in Computer Science, Vol. 3376.

[32] Kircanski, Aleksandar. Thesis: Cryptanalysis of Symmetric Cryptographic Primitives. 2013.

[33] De Cannière, Christophe, Lano, Joseph and Preneel, Bart. Comments on the Rediscovery of Time Memory Data Tradeoffs. 2005.

[34] Aumasson, Jean-Philippe. On a bias of Rabbit. 2006.

[35] Aumasson, Jean-Philippe, et al, Cube Testers and Key Recovery Attacks On Reduced-Round MD6 and Trivium. 2009.

[36]] Maximov, Alexander and Biryukov, Alex, Two Trivial Attacks on Trivium. 2007

[37] . Khovratovich, Dmitry, Rechberger, Christian and Savelieva, Alexandra. Bicliques for Preimages: Attacks on Skein-512 and the SHA-2 family. 2011.

[38] Lamberger, Mario and Mendel, Florian. Higher-Order Differential Attack on Reduced SHA-256. 2011.

[39] Mendel, Florian, et al .The Rebound Attack: Cryptanalysis of Reduced Whirlpool and Grøstl. 2009.

[40] Lamberger, Mario, et al. The Rebound Attack and Subspace Distinguishers: Application to Whirlpool. 2010.

[41] Sasaki, Yu. Meet-in-the-Middle Preimage Attacks on AES Hashing Modes and an Application to Whirlpool. 2011.

[42] . Lenstra, Arjen K. and Lenstra, Hendrik W. The development of the number field sieve. Springer, 1993.

[43] Lenstra, Hendrik Willem. Factoring integers with elliptic curves. 1987.

[44] Coppersmith, Don. Small Solutions to Polynomial Equations, and Low Exponent RSA Vulnerabilities. Journal of Cryptology, 1997.

[45] Shor, Peter. W Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer. 1994.

[46] Coppersmith, Don. Finding a small root of a bivariate integer equation; Factoring with high bits known. 1996.

[47] RFC 4871.

[48] Boneh, Dan and Durfee, Glenn. Cryptanalysis of RSA with private key d less than $2^{0.292}$. 2000.