TCG Algorithm Registry

Family “2.0"

Level 00 Revision 01.27

February 7, 2018

Contact: admin@trustedcomputinggroup.org

TCG Published

TCG

Disclaimers, Notices, and License Terms

Change History

Version	Date	Description
1.27	09.26.2017	Added change history. CMAC algorithm assigned as 0x003F in Table 3. Removed extra whitespace above the title of clause 6.10. Assigned bitfields in Table 21 for SHA3_256, SHA3_384 and SHA3_512. Replaced references of deprecated IETF 3447 with IETF 8017. Removed Annex A (Applicability of this Registry for Other TCG Specifications)

CONTENTS

TCG Algorithm Registry

1 Introduction

The Algorithm Registry lists each algorithm assigned an identifier, allowing it to be unambiguously defined and referenced by other TCG specifications. This document is a compendium of data related to the various algorithms used in specifications created by the Trusted Computing Group (TCG). The compendium of algorithm data is intended to ensure interoperability between devices built to be compliant with TCG specifications.

Many TCG specifications use a layered architecture where a single "library" specification on a bottom layer may be used by numerous platform specific middle layers (e.g. PC Client or Mobile Platform) to enable a variety of top level use cases. TCG specifications support products and solutions for numerous markets with varied requirements for commercial usefulness including features, security, interoperability, globalization, performance, regulatory requirements, compatibility, compliance, intellectual property rights, certification, etc. TCG as an organization does not perform cryptographic analysis of algorithms. The presence of an algorithm in the registry does not endorse its use by TCG for any specific use case or indicate an algorithm's acceptability for meeting any particular requirement set. The TCG endeavors to provide a variety of algorithms of varying strength for various commercial purposes. Ultimately, the TCG adds algorithms to its registry based on the needs of its membership.

Security is built into an increasing number of general purpose Information and Communications Technology (ICT) products, and security standards are fundamental to the integrity and sustainability of the global ICT infrastructure. The Trusted Computing Group (TCG) believes that open, interoperable, and internationally vetted standards are critical for the success of trusted computing, and that the multilateral approach to creating such standards is most effective.

TCG recognizes international standards in the field of IT security as the most appropriate method to ensure efficacy, interoperability, adoption and user acceptance. TCG takes into consideration international market requirements through international membership and welcomes participation from industry, academia, and governments in a unified, worldwide Trusted Computing standards development process.

Commercial implementation of TCG standards is managed by individual product and service providers. Implementers or adopters of any solution using TCG specifications must carefully assess the appropriateness of any algorithms or TCG specification for satisfying their goals. In assessing algorithms, TCG recommends implementers and adopters diligently evaluate available information such as governmental, industrial, and academic research. Solutions involving cryptography are dependent on the solution architecture and on the properties of cryptographic algorithms supported. Over time, cryptographic algorithms can develop deficiencies for reasons like advances in cryptographic techniques or increased computing power. Solutions that support a diversity of algorithms can remain durable when subsets of supported algorithms wane in usefulness. Therefore, implementers intent on providing robust solutions are responsible for evaluating both algorithm appropriateness and diversity.

The TCG classifies algorithms listed in this registry according to the following labels:

- **TCG Standard** - The algorithm is mandatory in one or more TCG specifications that reference this registry. The TCG designates algorithms with this classification in accordance with its goals of promoting international standards and interoperability.
- **TCG Legacy** – The algorithm is assigned an identifier for compatibility or historical reasons and is unlikely to be referenced by future TCG specifications. The TCG designates an algorithm with this classification based on the goals of the organization to discontinue support for the algorithm and transition solutions to alternative algorithms. Stakeholders using solutions relying on algorithms classified as TCG Legacy are strongly recommended to reevaluate the algorithm's appropriateness based on the current state of the art.

- **Assigned** – The algorithm is assigned an identifier, allowing it to be unambiguously defined and referenced by other TCG specifications, but is not designated as TCG Standard or TCG Legacy.

In terms of algorithm lifecycle in the registry, the TCG will initially assign algorithms to the Assigned classification. Some algorithms will be reclassified as TCG Standard if they become mandatory algorithms in TCG specifications. Eventually, algorithms are expected to transition to the TCG Legacy categorization.

2 Conventions

2.1 Bit and Octet Numbering and Order

An integer value is considered to be an array of one or more octets. The octet at offset zero within the array is the most significant octet (MSO) of the integer. Bit number 0 of that integer is its least significant bit and is the least significant bit in the last octet in the array.

EXAMPLE A 32-bit integer is an array of four octets; the MSO is at offset [0], and the most significant bit is bit number 31. Bit zero of this 32-bit integer is the least significant bit in the octet at offset [3] in the array.

NOTE Array indexing is zero-based.

The first listed member of a structure is at the lowest offset within the structure and the last listed member is at the highest offset within the structure.

For a character string (letters delimited by “”), the first character of the string contains the MSO.

2.2 Sized Buffer References

The specification makes extensive use of a data structure called a *sized buffer*. A sized buffer has a size field followed by an array of octets equal in number to the value in the size field.

The structure will have an identifying name. When the specification references the size field of the structure, the structure name is followed by “.size” (a period followed by the word “size”). When the specification references the octet array of the structure, the structure name is followed by “.buffer” (a period followed by the word “buffer”).

2.3 Numbers

Numbers are decimal unless a different radix is indicated.

Unless the number appears in a table intended to be machine readable, the radix is a subscript following the digits of the number. Only radix values of 2 and 16 are used in this specification.

Radix 16 (hexadecimal) numbers have a space separator between groups of two hexadecimal digits.

EXAMPLE 1 $40\ FF\ 12\ 34_{16}$

Radix 2 (binary) numbers use a space separator between groups of four binary digits.

EXAMPLE 2 $0100\ 1110\ 0001_2$

For numbers using a binary radix, the number of digits indicates the number of bits in the representation.

EXAMPLE 3 20_{16} is a hexadecimal number that contains exactly 8 bits and has a decimal value of 32.

EXAMPLE 4 $10\ 0000_2$ is a binary number that contains exactly 6 bits and has a decimal value of 32.

EXAMPLE 5 $0\ 20_{16}$ is a hexadecimal number that contains exactly 12 bits and has a decimal value of 32.

A number in a machine-readable table may use the “0x” prefix to denote a base 16 number. In this format, the number of digits is not always indicative of the number of bits in the representation.

EXAMPLE 6 0x20 is a hexadecimal number with a value of 32, and the number of bits is determined by the context.

3 Notation

The notations in this clause describe the representation of various data so that it is both human readable and amenable to automated processing.

3.1 Named Constants

A named constant is a numeric value to which a name has been assigned. In the C language, this is done with a `#define` statement. In this specification, a named constant is defined in a table that has a title that starts with “Definition” and ends with “Constants.”

The table title will indicate the name of the class of constants that are being defined in the table. The title will include the data type of the constants in parentheses.

The table in Example 1 names a collection of 16-bit constants.

EXAMPLE 1

Table xx — Definition of (UINT16) COUNTING Constants

Parameter	Value	Description
first	1	decimal value is implicitly the size of the
second	0x0002	hex value will match the number of bits in the constant
third	3	
fourth	0x0004	

3.2 Enumerations

A table that defines an enumerated data type will start with the word “Definition” and end with “Values.”

A value in parenthesis will denote the intrinsic data size of the value and may have the values "INT8", "UINT8", "INT16", “UINT16”, "INT32", and “UINT32.” If this value is not present, “UINT16” is assumed.

The table in Example 1 shows how an enumeration would be defined in this specification.

EXAMPLE 1

Table xx — Definition of (UINT16) CARD_SUIT Values

Suit Names	Value	Description
CLUBS	0x0000	
DIAMONDS	0x000D	
HEARTS	0x001A	
SPADES	0x0027	

3.3 Bit Field Definitions

A table that defines a structure containing bit fields has a title that starts with "Definition" and ends with "Bits." A type identifier in parentheses in the title indicates the size of the datum that contains the bit fields.

When the bit fields do not occupy consecutive locations, a spacer field is defined with a name of "Reserved." Bits in these spaces are reserved and shall be zero.

The table in Example 1 shows how a structure containing bit fields would be defined in this specification.

When a field has more than one bit, the range is indicated by a pair of numbers separated by a colon (":"). The numbers will be in high:low order.

EXAMPLE1

Table xx — Definition of (UINT32) SOME_ATTRIBUTE Bits

Bit	Name	Action
0	zeroth_bit	**SET (1)**: what to do if bit is 1 **CLEAR (0):** what to do if bit is 0
1	first_bit	**SET (1)**: what to do if bit is 1 **CLEAR (0):** what to do if bit is 0
6:2	Reserved	A placeholder that spans 5 bits
7	third_bit	**SET (1)**: what to do if bit is 1 **CLEAR (0):** what to do if bit is 0
31:8	Reserved	Placeholder to fill 32 bits

3.4 Name Prefix Convention

Parameters are constants, variables, structures, unions, and structure members. Structure members are given a name that is indicative of its use, with no special prefix. The other parameter types are named according to their type with their name starting with "TPMx_", where "x" is an optional character to indicate the data type.

In some cases, additional qualifying characters will follow the underscore. These are generally used when dealing with an enumerated data type.

Table 1 — Name Prefix Convention

Prefix	Description
TPM_	a constant or an enumerated type
TPM_ALG_	an enumerated type that indicates an algorithm A TPM_ALG_ is often used as a selector for a union.
TPM_xx_	an enumeration value of a particular type The value of "xx" will be indicative of the use of the enumerated type. A table of "TPM_xx" constant definitions will exist to define each of the TPM_xx_ values. EXAMPLE 1 TPM_RC_ indicates that the type is used for a *responseCode*.

4 TPM_ALG_ID

Table 3 is the list of algorithms to which the TCG has assigned an algorithm identifier along with its numeric identifier.

An algorithm ID is often used like a tag to determine the type of a structure in a context-sensitive way. The values for TPM_ALG_ID shall be in the range of 00 00_{16} – 7F FF_{16}. Other structure tags will be in the range 80 00_{16} – FF FF_{16}.

An algorithm shall not be assigned a value in the range 00 $C1_{16}$ – 00 $C6_{16}$ in order to prevent any overlap with the command structure tags used in TPM 1.2.

The implementation of some algorithms is dependent on the presence of other algorithms. When there is a dependency, the algorithm that is required is listed in column labeled "Dep" (Dependent) in Table 4.

EXAMPLE Implementation of TPM_ALG_RSASSA requires that the RSA algorithm be implemented.

TPM_ALG_KEYEDHASH and TPM_ALG_NULL are required of all TPM implementations.

Table 2 — Legend for TPM_ALG_ID Table

Column Title	**Comments**
Algorithm Name	the mnemonic name assigned to the algorithm
Value	the numeric value assigned to the algorithm
Type	The allowed values are: **A** – asymmetric algorithm with a public and private key **S** – symmetric algorithm with only a private key **H** – hash algorithm that compresses input data to a digest value or indicates a method that uses a hash **X** – signing algorithm **N** – an anonymous signing algorithm **E** – an encryption mode **M** – a method such as a mask generation function **O** – an object type
C	(**C**lassification) The allowed values are: **A** – Assigned **S** – TCG Standard **L** – TCG Legacy
Dep	(**D**ependent) Indicates which other algorithm is required to be implemented if this algorithm is implemented
Reference	the reference document that defines the algorithm
Comments	clarifying information

Table 3 — Definition of (UINT16) TPM_ALG_ID Constants

Algorithm Name	Value	Type	Dep	C	Reference	Comments
TPM_ALG_ERROR	0x0000					should not occur
TPM_ALG_RSA	0x0001	A O		S	IETF RFC 8017	the RSA algorithm
TPM_ALG_TDES	0x0003	S		A	ISO/IEC 18033-3	block cipher with various key sizes (Triple Data Encryption Algorithm, commonly called Triple Data Encryption Standard)
TPM_ALG_SHA	0x0004	H		S	ISO/IEC 10118-3	the SHA1 algorithm
TPM_ALG_SHA1	0x0004	H		S	ISO/IEC 10118-3	redefinition for documentation consistency
TPM_ALG_HMAC	0x0005	H X		S	ISO/IEC 9797-2	Hash Message Authentication Code (HMAC) algorithm
TPM_ALG_AES	0x0006	S		S	ISO/IEC 18033-3	the AES algorithm with various key sizes
TPM_ALG_MGF1	0x0007	H M		S	IEEE Std 1363™-2000 IEEE Std 1363a™-2004	hash-based mask-generation function
TPM_ALG_KEYEDHASH	0x0008	H O		S	TCG TPM 2.0 library specification	an object type that may use XOR for encryption or an HMAC for signing and may also refer to a data object that is neither signing nor encrypting
TPM_ALG_XOR	0x000A	H S		S	TCG TPM 2.0 library specification	the XOR encryption algorithm
TPM_ALG_SHA256	0x000B	H		S	ISO/IEC 10118-3	the SHA 256 algorithm
TPM_ALG_SHA384	0x000C	H		A	ISO/IEC 10118-3	the SHA 384 algorithm
TPM_ALG_SHA512	0x000D	H		A	ISO/IEC 10118-3	the SHA 512 algorithm
TPM_ALG_NULL	0x0010			S	TCG TPM 2.0 library specification	Null algorithm
TPM_ALG_SM3_256	0x0012	H		A	GM/T 0004-2012	SM3 hash algorithm
TPM_ALG_SM4	0x0013	S		A	GM/T 0002-2012	SM4 symmetric block cipher
TPM_ALG_RSASSA	0x0014	A X	RSA	S	IETF RFC 8017	a signature algorithm defined in section 8.2 (RSASSA-PKCS1-v1_5)
TPM_ALG_RSAES	0x0015	A E	RSA	S	IETF RFC 8017	a padding algorithm defined in section 7.2 (RSAES-PKCS1-v1_5)
TPM_ALG_RSAPSS	0x0016	A X	RSA	S	IETF RFC 8017	a signature algorithm defined in section 8.1 (RSASSA-PSS)
TPM_ALG_OAEP	0x0017	A E H	RSA	S	IETF RFC 8017	a padding algorithm defined in section 7.1 (RSAES_OAEP)

Algorithm Name	Value	Type	Dep	C	Reference	Comments
TPM_ALG_ECDSA	0x0018	A X	ECC	S	ISO/IEC 14888-3	signature algorithm using elliptic curve cryptography (ECC)
TPM_ALG_ECDH	0x0019	A M	ECC	S	NIST SP800-56A	secret sharing using ECC Based on context, this can be either One-Pass Diffie-Hellman, C(1, 1, ECC CDH) defined in 6.2.2.2 or Full Unified Model C(2, 2, ECC CDH) defined in 6.1.1.2
TPM_ALG_ECDAA	0x001A	A X N	ECC	S	TCG TPM 2.0 library specification	elliptic-curve based, anonymous signing scheme
TPM_ALG_SM2	0x001B	A X	ECC	A	GM/T 0003.1–2012 GM/T 0003.2–2012 GM/T 0003.3–2012 GM/T 0003.5–2012	SM2 – depending on context, either an elliptic-curve based, signature algorithm or a key exchange protocol NOTE 1 Type listed as signing but, other uses are allowed according to context.
TPM_ALG_ECSCHNORR	0x001C	A X	ECC	S	TCG TPM 2.0 library specification	elliptic-curve based Schnorr signature
TPM_ALG_ECMQV	0x001D	A M	ECC	A	NIST SP800-56A	two-phase elliptic-curve key exchange – C(2, 2, ECC MQV) section 6.1.1.4
TPM_ALG_KDF1_SP800_56A	0x0020	H M	ECC	S	NIST SP800-56A	concatenation key derivation function (approved alternative 1) section 5.8.1
TPM_ALG_KDF2	0x0021	H M		A	IEEE Std 1363a-2004	key derivation function KDF2 section 13.2
TPM_ALG_KDF1_SP800_108	0x0022	H M		S	NIST SP800-108	a key derivation method Section 5.1 KDF in Counter Mode
TPM_ALG_ECC	0x0023	A O		S	ISO/IEC 15946-1	prime field ECC
TPM_ALG_SYMCIPHER	0x0025	O S		S	TCG TPM 2.0 library specification	the object type for a symmetric block cipher
TPM_ALG_CAMELLIA	0x0026	S		A	ISO/IEC 18033-3	Camellia is symmetric block cipher. The Camellia algorithm with various key sizes
TPM_ALG_SHA3_256	0x0027	H		A	NIST PUB FIPS 202	Hash algorithm producing a 256-bit digest
TPM_ALG_SHA3_384	0x0028	H		A	NIST PUB FIPS 202	Hash algorithm producing a 384-bit digest
TPM_ALG_SHA3_512	0x0029	H		A	NIST PUB FIPS 202	Hash algorithm producing a 512-bit digest

Algorithm Name	Value	Type	Dep	C	Reference	Comments
TPM_ALG_CMAC	0x003F	S X		A	ISO/IEC 9797-1:2011	block Cipher-based Message Authentication Code (CMAC) "Algorithm 5" in ISO/IEC 9797-1:2011
TPM_ALG_CTR	0x0040	S E		A	ISO/IEC 10116	Counter mode – if implemented, all symmetric block ciphers (S type) implemented shall be capable of using this mode.
TPM_ALG_OFB	0x0041	S E		A	ISO/IEC 10116	Output Feedback mode – if implemented, all symmetric block ciphers (S type) implemented shall be capable of using this mode.
TPM_ALG_CBC	0x0042	S E		A	ISO/IEC 10116	Cipher Block Chaining mode – if implemented, all symmetric block ciphers (S type) implemented shall be capable of using this mode.
TPM_ALG_CFB	0x0043	S E		S	ISO/IEC 10116	Cipher Feedback mode – if implemented, all symmetric block ciphers (S type) implemented shall be capable of using this mode.
TPM_ALG_ECB	0x0044	S E		A	ISO/IEC 10116	Electronic Codebook mode – if implemented, all symmetric block ciphers (S type) implemented shall be capable of using this mode. NOTE 2 This mode is not recommended for uses unless the key is frequently rotated such as in video codecs
reserved	0x00C1 through 0x00C6					0x00C1 – 0x00C6 are reserved to prevent any overlap with the command structure tags used in TPM 1.2
reserved	0x8000 through 0xFFFF					reserved for other structure tags

5 ECC Values

5.1 Curve ID Values

Table 4 is the list of identifiers for TCG-registered curve ID values for elliptic curve cryptography.

Table 4 — Definition of (UINT16) TPM_ECC_CURVE Constants

Name	Value	Classification	Comments
TPM_ECC_NONE	0x0000	Assigned	
TPM_ECC_NIST_P192	0x0001	Assigned	
TPM_ECC_NIST_P224	0x0002	Assigned	
TPM_ECC_NIST_P256	0x0003	TCG Standard	
TPM_ECC_NIST_P384	0x0004	Assigned	
TPM_ECC_NIST_P521	0x0005	Assigned	
TPM_ECC_BN_P256	0x0010	TCG Standard	curve to support ECDAA
TPM_ECC_BN_P638	0x0011	Assigned	curve to support ECDAA
TPM_ECC_SM2_P256	0x0020	Assigned	
#TPM_RC_CURVE			NOTE This row has meaning for other TCG specifications that use automated processing and should be ignored for the TCG Algorithm Registry.

5.2 Curve Parameters

5.2.1 Introduction

The tables in this section contain the curve parameter data associated with the curves listed in Table 4.

5.2.2 NIST P192

Table 5 — Defines for NIST_P192 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_NIST_P192	identifier for the curve
keySize	192	size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SHA256}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	**{24, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF ,0xFF, 0xFF}}**	*Fp* (the modulus)
a	**{24, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC}}**	coefficient of the linear term in the curve equation
b	**{24, {0x64, 0x21, 0x05, 0x19, 0xE5, 0x9C, 0x80, 0xE7, 0x0F, 0xA7, 0xE9, 0xAB, 0x72, 0x24, 0x30, 0x49, 0xFE, 0xB8, 0xDE, 0xEC, 0xC1, 0x46, 0xB9, 0xB1}}**	constant term for curve equation
gX	**{24, {0x18, 0x8D, 0xA8, 0x0E, 0xB0, 0x30, 0x90, 0xF6, 0x7C, 0xBF, 0x20, 0xEB, 0x43, 0xA1, 0x88, 0x00, 0xF4, 0xFF, 0x0A, 0xFD, 0x82, 0xFF, 0x10, 0x12}}**	x coordinate of base point G
gY	**{24, {0x07, 0x19, 0x2B, 0x95, 0xFFC, 0x8D, 0xA7, 0x86, 0x31, 0x01, 0x1ED, 0x6B, 0x24, 0xCD, 0xD5, 0x73, 0xF9, 0x77, 0xA1, 0x1E, 0x79, 0x48, 0x11}}**	y coordinate of base point G
n	**{24, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x99, 0xDE, 0xF8, 0x36, 0x14, 0x6B, 0xC9, 0xB1, 0xB4, 0xD2, 0x28, 0x31}}**	order of G
h	**{1,{1}}**	cofactor (a size of zero indicates a cofactor of 1)

5.2.3 NIST P224

Table 6 — Defines for NIST_P224 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_NIST_P224	identifier for the curve
keySize	224	Size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SHA256}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	{28, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x01 }}	Fp (the modulus)
a	{28, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFE }}	coefficient of the linear term in the curve equation
b	{28, {0xB4, 0x05, 0x0A, 0x85, 0x0C, 0x04, 0xB3, 0xAB, 0xF5, 0x41, 0x32, 0x56, 0x50, 0x44, 0xB0, 0xB7, 0xD7, 0xBF, 0xD8, 0xBA, 0x27, 0x0B, 0x39, 0x43, 0x23, 0x55, 0xFF, 0xB4 }}	constant term for curve equation
gX	{28, {0xB7, 0x0E, 0x0C, 0xBD, 0x6B, 0xB4, 0xBF, 0x7F, 0x32, 0x13, 0x90, 0xB9, 0x4A, 0x03, 0xC1, 0xD3, 0x56, 0xC2, 0x11, 0x22, 0x34, 0x32, 0x80, 0xD6, 0x11, 0x5C, 0x1D, 0x21 }}	x coordinate of base point G
gY	{28, {0xBD, 0x37, 0x63, 0x88, 0xB5, 0xF7, 0x23, 0xFB, 0x4C, 0x22, 0xDF, 0xE6, 0xCD, 0x43, 0x75, 0xA0, 0x5A, 0x07, 0x47, 0x64, 0x44, 0xD5, 0x81, 0x99, 0x85, 0x00, 0x7E, 0x34 }}	y coordinate of base point G
n	{28, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x16, 0xA2, 0xE0, 0xB8, 0xF0, 0x3E, 0x13, 0xDD, 0x29, 0x45, 0x5C, 0x5C, 0x2A, 0x3D }}	order of G
h	{1,{1}}	cofactor

5.2.4 NIST P256

Table 7 — Defines for NIST_P256 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_NIST_P256	identifier for the curve
keySize	256	Size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SHA256}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	{32, {0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF }}	*Fp* (the modulus)
a	{32, {0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x01, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC }}	coefficient of the linear term in the curve equation
b	{32, {0x5A, 0xC6, 0x35, 0xD8, 0xAA, 0x3A, 0x93, 0xE7, 0xB3, 0xEB, 0xBD, 0x55, 0x76, 0x98, 0x86, 0xBC, 0x65, 0x1D, 0x06, 0xB0, 0xCC, 0x53, 0xB0, 0xF6, 0x3B, 0xCE, 0x3C, 0x3E, 0x27, 0xD2, 0x60, 0x4B }}	constant term for curve equation
gX	{32, {0x6B, 0x17, 0xD1, 0xF2, 0xE1, 0x2C, 0x42, 0x47, 0xF8, 0xBC, 0xE6, 0xE5, 0x63, 0xA4, 0x40, 0xF2, 0x77, 0x03, 0x7D, 0x81, 0x2D, 0xEB, 0x33, 0xA0, 0xF4, 0xA1, 0x39, 0x45, 0xD8, 0x98, 0xC2, 0x96 }}	x coordinate of base point G
gY	{32, {0x4F, 0xE3, 0x42, 0xE2, 0xFE, 0x1A, 0x7F, 0x9B, 0x8E, 0xE7, 0xEB, 0x4A, 0x7C, 0x0F, 0x9E, 0x16, 0x2B, 0xCE, 0x33, 0x57, 0x6B, 0x31, 0x5E, 0xCE, 0xCB, 0xB6, 0x40, 0x68, 0x37, 0xBF, 0x51, 0xF5 }}	y coordinate of base point G
n	{32, {0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xBC, 0xE6, 0xFA, 0xAD, 0xA7, 0x17, 0x9E, 0x84, 0xF3, 0xB9, 0xCA, 0xC2, 0xFC, 0x63, 0x25, 0x51 }}	order of G
h	{1,{1}}	cofactor

5.2.5 NIST P384

Table 8 — Defines for NIST_P384 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_NIST_P384	identifier for the curve
keySize	384	size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SHA384}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	`{48, {0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF }}`	*Fp* (the modulus)
a	`{48, {0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFC }}`	coefficient of the linear term in the curve equation
b	`{48, {0xB3, 0x31, 0x2F, 0xA7, 0xE2, 0x3E, 0xE7, 0xE4, 0x98, 0x8E, 0x05, 0x6B, 0xE3, 0xF8, 0x2D, 0x19, 0x18, 0x1D, 0x9C, 0x6E, 0xFE, 0x81, 0x41, 0x12, 0x03, 0x14, 0x08, 0x8F, 0x50, 0x13, 0x87, 0x5A, 0xC6, 0x56, 0x39, 0x8D, 0x8A, 0x2E, 0xD1, 0x9D, 0x2A, 0x85, 0xC8, 0xED, 0xD3, 0xEC, 0x2A, 0xEF }}`	constant term for curve equation
gX	`{48, {0xAA, 0x87, 0xCA, 0x22, 0xBE, 0x8B, 0x05, 0x37, 0x8E, 0xB1, 0xC7, 0x1E, 0xF3, 0x20, 0xAD, 0x74, 0x6E, 0x1D, 0x3B, 0x62, 0x8B, 0xA7, 0x9B, 0x98, 0x59, 0xF7, 0x41, 0xE0, 0x82, 0x54, 0x2A, 0x38, 0x55, 0x02, 0xF2, 0x5D, 0xBF, 0x55, 0x29, 0x6C, 0x3A, 0x54, 0x5E, 0x38, 0x72, 0x76, 0x0A, 0xB7 }}`	x coordinate of base point G
gY	`{48, {0x36, 0x17, 0xDE, 0x4A, 0x96, 0x26, 0x2C, 0x6F, 0x5D, 0x9E, 0x98, 0xBF, 0x92, 0x92, 0xDC, 0x29, 0xF8, 0xF4, 0x1D, 0xBD, 0x28, 0x9A, 0x14, 0x7C, 0xE9, 0xDA, 0x31, 0x13, 0xB5, 0xF0, 0xB8, 0xC0, 0x0A, 0x60, 0xB1, 0xCE, 0x1D, 0x7E, 0x81, 0x9D, 0x7A, 0x43, 0x1D, 0x7C, 0x90, 0xEA, 0x0E, 0x5F }}`	y coordinate of base point G
n	`{48, {0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xC7, 0x63, 0x4D, 0x81, 0xF4, 0x37, 0x2D, 0xDF, 0x58, 0x1A, 0x0D, 0xB2, 0x48, 0xB0, 0xA7, 0x7A, 0xEC, 0xEC, 0x19, 0x6A, 0xCC, 0xC5, 0x29, 0x73 }}`	order of G
h	`{1,{1}}`	cofactor

5.2.6 NIST P521

Table 9 — Defines for NIST_P521 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_NIST_P521	identifier for the curve
keySize	521	size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SHA512}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme

p	`{66, {0x01, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF }}`
		Fp (the modulus)

a	`{66, {0x01, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC }}`
		coefficient of the linear term in the curve equation

b	`{66, {0x00, 0x51, 0x95, 0x3E, 0xB9, 0x61, 0x8E, 0x1C, 0x9A,`
	`0x1F, 0x92, 0x9A, 0x21, 0xA0, 0xB6, 0x85, 0x40, 0xEE, 0xA2,`
	`0xDA, 0x72, 0x5B, 0x99, 0xB3, 0x15, 0xF3, 0xB8, 0xB4, 0x89,`
	`0x91, 0x8E, 0xF1, 0x09, 0xE1, 0x56, 0x19, 0x39, 0x51, 0xEC,`
	`0x7E, 0x93, 0x7B, 0x16, 0x52, 0xC0, 0xBD, 0x3B, 0xB1, 0xBF,`
	`0x07, 0x35, 0x73, 0xDF, 0x88, 0x3D, 0x2C, 0x34, 0xF1, 0xEF,`
	`0x45, 0x1F, 0xD4, 0x6B, 0x50, 0x3F, 0x00 }}`
		constant term for curve equation

gX	`{66, {0x00, 0xC6, 0x85, 0x8E, 0x06, 0xB7, 0x04, 0x04, 0xE9,`
	`0xCD, 0x9E, 0x3E, 0xCB, 0x66, 0x23, 0x95, 0xB4, 0x42, 0x9C,`
	`0x64, 0x81, 0x39, 0x05, 0x3F, 0xB5, 0x21, 0xF8, 0x28, 0xAF,`
	`0x60, 0x6B, 0x4D, 0x3D, 0xBA, 0xA1, 0x4B, 0x5E, 0x77, 0xEF,`
	`0xE7, 0x59, 0x28, 0xFE, 0x1D, 0xC1, 0x27, 0xA2, 0xFF, 0xA8,`
	`0xDE, 0x33, 0x48, 0xB3, 0xC1, 0x85, 0x6A, 0x42, 0x9B, 0xF9,`
	`0x7E, 0x7E, 0x31, 0xC2, 0xE5, 0xBD, 0x66 }}`
		x coordinate of base point G

gY	`{66, {0x01, 0x18, 0x39, 0x29, 0x6A, 0x78, 0x9A, 0x3B, 0xC0,`
	`0x04, 0x5C, 0x8A, 0x5F, 0xB4, 0x2C, 0x7D, 0x1B, 0xD9, 0x98,`
	`0xF5, 0x44, 0x49, 0x57, 0x9B, 0x44, 0x68, 0x17, 0xAF, 0xBD,`
	`0x17, 0x27, 0x3E, 0x66, 0x2C, 0x97, 0xEE, 0x72, 0x99, 0x5E,`
	`0xF4, 0x26, 0x40, 0xC5, 0x50, 0xB9, 0x01, 0x3F, 0xAD, 0x07,`
	`0x61, 0x35, 0x3C, 0x70, 0x86, 0xA2, 0x72, 0xC2, 0x40, 0x88,`
	`0xBE, 0x94, 0x76, 0x9F, 0xD1, 0x66, 0x50 }}`
		y coordinate of base point G

n	`{66, {0x01, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF,`
	`0xFF, 0xFF, 0xFF, 0xFF, 0xFA, 0x51, 0x86, 0x87, 0x83, 0xBF,`
	`0x2F, 0x96, 0x6B, 0x7F, 0xCC, 0x01, 0x48, 0xF7, 0x09, 0xA5,`
	`0xD0, 0x3B, 0xB5, 0xC9, 0xB8, 0x89, 0x9C, 0x47, 0xAE, 0xBB,`
	`0x6F, 0xB7, 0x1E, 0x91, 0x38, 0x64, 0x09 }}`
		order of G

h	`{1,{1}}`	cofactor

5.2.7 BN P256

Table 10 — Defines for BN_P256 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_BN_P256	identifier for the curve
keySize	256	size in bits of the key
kdf	{TPM_ALG_NULL, TPM_ALG_NULL}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	`{32, {0xFF, 0XFF, 0XFF, 0XFF, 0XFF, 0XFC, 0XF0, 0XCD, 0X46, 0XE5, 0XF2, 0X5E, 0XEE, 0X71, 0XA4, 0X9F, 0X0C, 0XDC, 0X65, 0XFB, 0X12, 0X98, 0X0A, 0X82, 0XD3, 0X29, 0X2D, 0XDB, 0XAE, 0XD3, 0X30, 0X13 }}`	*Fp* (the modulus)
a	`{1,{0}}`	coefficient of the linear term in the curve equation
b	`{1,{3}}`	constant term for curve equation
gX	`{1,{1}}`	x coordinate of base point G
gY	`{1,{2}};`	y coordinate of base point G
n	`{32, {0xFF, 0XFF, 0XFF, 0XFF, 0XFF, 0XFC, 0XF0, 0XCD, 0X46, 0XE5, 0XF2, 0X5E, 0XEE, 0X71, 0XA4, 0X9E, 0X0C, 0XDC, 0X65, 0XFB, 0X12, 0X99, 0X92, 0X1A, 0XF6, 0X2D, 0X53, 0X6C, 0XD1, 0X0B, 0X50, 0X0D }}`	order of G
h	`{1,{1}}`	cofactor

5.2.8 BN P638

Table 11 — Defines for BN_P638 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_BN_P638	identifier for the curve
keySize	638	size in bits of the key
kdf	{TPM_ALG_NULL, TPM_ALG_NULL}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	`{80, {0x23, 0xFF, 0xFF, 0xFD, 0xC0, 0x00, 0x00, 0x0D, 0x7F, 0xFF, 0xFF, 0xB8, 0x00, 0x00, 0x01, 0xD3, 0xFF, 0xFF, 0xF9, 0x42, 0xD0, 0x00, 0x16, 0x5E, 0x3F, 0xFF, 0x94, 0x87, 0x00, 0x00, 0xD5, 0x2F, 0xFF, 0xFD, 0xD0, 0xE0, 0x00, 0x08, 0xDE, 0x55, 0xC0, 0x00, 0x86, 0x52, 0x00, 0x21, 0xE5, 0x5B, 0xFF, 0xFF, 0xF5, 0x1F, 0xFF, 0xF4, 0xEB, 0x80, 0x00, 0x00, 0x00, 0x4C, 0x80, 0x01, 0x5A, 0xCD, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xEC, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x67 }}`	*Fp* (the modulus)
a	`{1,{0}}`	coefficient of the linear term in the curve equation
b	`{2,{0x01, 0x01}}`	constant term for curve equation
gX	`{80, {0x23, 0xFF, 0xFF, 0xFD, 0xC0, 0x00, 0x00, 0x0D, 0x7F, 0xFF, 0xFF, 0xB8, 0x00, 0x00, 0x01, 0xD3, 0xFF, 0xFF, 0xF9, 0x42, 0xD0, 0x00, 0x16, 0x5E, 0x3F, 0xFF, 0x94, 0x87, 0x00, 0x00, 0xD5, 0x2F, 0xFF, 0xFD, 0xD0, 0xE0, 0x00, 0x08, 0xDE, 0x55, 0xC0, 0x00, 0x86, 0x52, 0x00, 0x21, 0xE5, 0x5B, 0xFF, 0xFF, 0xF5, 0x1F, 0xFF, 0xF4, 0xEB, 0x80, 0x00, 0x00, 0x00, 0x4C, 0x80, 0x01, 0x5A, 0xCD, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xEC, 0xE0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x66 }}`	x coordinate of base point G
gY	`{1,{0x10}}`	y coordinate of base point G
n	`{80, {0x23, 0xFF, 0xFF, 0xFD, 0xC0, 0x00, 0x00, 0x0D, 0x7F, 0xFF, 0xFF, 0xB8, 0x00, 0x00, 0x01, 0xD3, 0xFF, 0xFF, 0xF9, 0x42, 0xD0, 0x00, 0x16, 0x5E, 0x3F, 0xFF, 0x94, 0x87, 0x00, 0x00, 0xD5, 0x2F, 0xFF, 0xFD, 0xD0, 0xE0, 0x00, 0x08, 0xDE, 0x55, 0x60, 0x00, 0x86, 0x55, 0x00, 0x21, 0xE5, 0x55, 0xFF, 0xFF, 0xF5, 0x4F, 0xFF, 0xF4, 0xEA, 0xC0, 0x00, 0x00, 0x00, 0x49, 0x80, 0x01, 0x54, 0xD9, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xED, 0xA0, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x00, 0x61 }}`	order of G
h	`{1,{1}}`	cofactor

5.2.9 SM2_P256

Table 12 — Defines for SM2_P256 ECC Values

Parameter	Value	Description
curveID	TPM_ECC_SM2_P256	identifier for the curve
keySize	256	size in bits of the key
kdf	{TPM_ALG_KDF1_SP800_56A, TPM_ALG_SM3_256}	the default KDF and hash
sign	{TPM_ALG_NULL, TPM_ALG_NULL}	no mandatory signing scheme
p	**{32, {0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF }}**	*Fp* (the modulus)
a	**{32, {0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x00, 0x00, 0x00, 0x00, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFC }}**	coefficient of the linear term in the curve equation
b	**{32, {0x28, 0xE9, 0xFA, 0x9E, 0x9D, 0x9F, 0x5E, 0x34, 0x4D, 0x5A, 0x9E, 0x4B, 0xCF, 0x65, 0x09, 0xA7, 0xF3, 0x97, 0x89, 0xF5, 0x15, 0xAB, 0x8F, 0x92, 0xDD, 0xBC, 0xBD, 0x41, 0x4D, 0x94, 0x0E, 0x93 }}**	constant term for curve equation
gX	**{32, {0x32, 0xC4, 0xAE, 0x2C, 0x1F, 0x19, 0x81, 0x19, 0x5F, 0x99, 0x04, 0x46, 0x6A, 0x39, 0xC9, 0x94, 0x8F, 0xE3, 0x0B, 0xBF, 0xF2, 0x66, 0x0B, 0xE1, 0x71, 0x5A, 0x45, 0x89, 0x33, 0x4C, 0x74, 0xC7 }}**	x coordinate of base point G
gY	**{32, {0xBC, 0x37, 0x36, 0xA2, 0xF4, 0xF6, 0x77, 0x9C, 0x59, 0xBD, 0xCE, 0xE3, 0x6B, 0x69, 0x21, 0x53, 0xD0, 0xA9, 0x87, 0x7C, 0xC6, 0x2A, 0x47, 0x40, 0x02, 0xDF, 0x32, 0xE5, 0x21, 0x39, 0xF0, 0xA0 }}**	y coordinate of base point G
n	**{32, {0xFF, 0xFF, 0xFF, 0xFE, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0xFF, 0x72, 0x03, 0xDF, 0x6B, 0x21, 0xC6, 0x05, 0x2B, 0x53, 0xBB, 0xF4, 0x09, 0x39, 0xD5, 0x41, 0x23 }}**	order of G
H	**{1,{1}}**	cofactor

6 Hash Parameters

6.1 Introduction

The tables in this clause define the basic parameters associated with the TCG-registered hash algorithms listed in Table 3.

6.2 SHA1

Table 13 — Defines for SHA1 Hash Values

Name	Value	Description
SHA1_DIGEST_SIZE	20	size of digest in octets
SHA1_BLOCK_SIZE	64	size of hash block in octets
SHA1_DER_SIZE	15	size of the DER in octets
SHA1_DER	`0x30, 0x21, 0x30, 0x09, 0x06, 0x05, 0x2B, 0x0E, 0x03, 0x02, 0x1A, 0x05, 0x00, 0x04, 0x14`	the DER

6.3 SHA256

Table 14 — Defines for SHA256 Hash Values

Name	Value	Description
SHA256_DIGEST_SIZE	32	size of digest
SHA256_BLOCK_SIZE	64	size of hash block
SHA256_DER_SIZE	19	size of the DER in octets
SHA256_DER	`0x30, 0x31, 0x30, 0x0d, 0x06, 0x09, 0x60, 0x86, 0x48, 0x01, 0x65, 0x03, 0x04, 0x02, 0x01, 0x05, 0x00, 0x04, 0x20`	the DER

6.4 SHA384

Table 15 — Defines for SHA384 Hash Values

Name	Value	Description
SHA384_DIGEST_SIZE	48	size of digest in octets
SHA384_BLOCK_SIZE	128	size of hash block in octets
SHA384_DER_SIZE	19	size of the DER in octets
SHA384_DER	`0x30, 0x41, 0x30, 0x0d, 0x06, 0x09, 0x60, 0x86, 0x48, 0x01, 0x65, 0x03, 0x04, 0x02, 0x02, 0x05, 0x00, 0x04, 0x30`	the DER

6.5 SHA512

Table 16 — Defines for SHA512 Hash Values

Name	Value	Description
SHA512_DIGEST_SIZE	64	size of digest in octets
SHA512_BLOCK_SIZE	128	size of hash block in octets
SHA512_DER_SIZE	19	size of the DER in octets
SHA512_DER	`0x30, 0x51, 0x30, 0x0d, 0x06, 0x09, 0x60, 0x86, 0x48, 0x01, 0x65, 0x03, 0x04, 0x02, 0x03, 0x05, 0x00, 0x04, 0x40`	the DER

6.6 SM3_256

Table 17 — Defines for SM3_256 Hash Values

Name	Value	Description
SM3_256_DIGEST_SIZE	32	size of digest in octets
SM3_256_BLOCK_SIZE	64	size of hash block in octets
SM3_256_DER_SIZE	18	size of the DER in octets
SM3_256_DER	`0x30, 0x30, 0x30, 0x0c, 0x06, 0x08, 0x2A, 0x81, 0x1C, 0x81, 0x45, 0x01, 0x83, 0x11, 0x05, 0x00, 0x04, 0x20`	the DER

6.7 SHA3_256

Table 18— Defines for SHA3_256 Hash Values

Name	Value	Description
SHA3_256_DIGEST_SIZE	32	size of digest in octets
SHA3_256_BLOCK_SIZE	136	size of hash block in octets
SHA3_256_DER_SIZE	19	size of the DER in octets
SHA3_256_DER	`0x30 0x31 0x30 0x0d 0x06 0x09 0x60 0x86 0x48 0x01 0x65 0x03 0x04 0x02 0x08 0x05 0x00 0x04 0x20`	the DER

6.8 SHA3_384

Table 19 — Defines for SHA3_384 Hash Values

Name	Value	Description
SHA3_384_DIGEST_SIZE	48	size of digest in octets
SHA3_384_BLOCK_SIZE	104	size of hash block in octets
SHA3_384_DER_SIZE	19	size of the DER in octets
SHA3_384_DER	0x30 0x41 0x30 0x0d 0x06 0x09 0x60 0x86 0x48 0x01 0x65 0x03 0x04 0x02 0x09 0x05 0x00 0x04 0x30	the DER

6.9 SHA3_512

Table 20 — Defines for SHA3_512 Hash Values

Name	Value	Description
SHA3_512_DIGEST_SIZE	64	size of digest in octets
SHA3_512_BLOCK_SIZE	72	size of hash block in octets
SHA3_512_DER_SIZE	19	size of the DER in octets
SHA3_512_DER	0x30 0x51 0x30 0x0d 0x06 0x09 0x60 0x86 0x48 0x01 0x65 0x03 0x04 0x02 0x0a 0x05 0x00 0x04 0x40	the DER

6.10 Hash Algorithms Bit Field

This table defines a bit field to concisely convey a set of hash algorithms. An example of where this could be useful is a parameter returning the set of hash algorithms an interface supports.

Table 21 — Definition of (UINT32) TPMA_HASH_ALGS Bits

Bit	Name	Action
0	hashAlgSHA1	**SET (1)**: indicates the SHA1 hash algorithm **CLEAR (0):** does not indicate SHA1
1	hashAlgSHA256	**SET (1)**: indicates the SHA256 hash algorithm **CLEAR (0):** does not indicate SHA256
2	hashAlgSHA384	**SET (1)**: indicates the SHA384 hash algorithm **CLEAR (0):** does not indicate SHA384
3	hashAlgSHA512	**SET (1)**: indicates the SHA512 hash algorithm **CLEAR (0):** does not indicate SHA512
4	hashAlgSM3_256	**SET (1)**: indicates the SM3_256 hash algorithm **CLEAR (0):** does not indicate SM3_256
5	hashAlgSHA3_256	**SET (1)**: indicates the SHA3_256 hash algorithm **CLEAR (0):** does not indicate SHA3_256
6	hashAlgSHA3_384	**SET (1)**: indicates the SHA3_384 hash algorithm **CLEAR (0):** does not indicate SHA3_384
7	hashAlgSHA3_512	**SET (1)**: indicates the SHA3_512 hash algorithm **CLEAR (0):** does not indicate SHA3_512
31:8	Reserved	Shall be zero

7 Symmetric Block Cipher Parameters

7.1 Introduction

The tables in this section define the parameters for each of the TCG-registered block ciphers listed in Table 3.

7.2 AES

Table 22 — Defines for AES Symmetric Cipher Algorithm Constants

Name	Value	Comments
AES_KEY_SIZES_BITS	{128, 192, 256}	
AES_BLOCK_SIZES_BITS	{128, 128, 128}	
AES_ROUNDS	{10, 12, 14}	

7.3 SM4

Table 23 — Defines for SM4 Symmetric Cipher Algorithm Constants

Name	Value	Comments
SM4_KEY_SIZES_BITS	{128}	
SM4_BLOCK_SIZES_BITS	{128}	
SM4_ROUNDS	{32}	

7.4 Camellia

Table 24 — Defines for CAMELLIA Symmetric Cipher Algorithm Constants

Name	Value	Comments
CAMELLIA_KEY_SIZES_BITS	{128, 192, 256}	
CAMELLIA_BLOCK_SIZES_BITS	{128, 128, 128}	the block size is the same for all key sizes
CAMELLIA_ROUNDS	{18, 24, 24}	

7.5 TDES

Definitions for two and three key triple-DES.

A TCG compliant device shall not allow a triple DES key to be used if K1 = K2, or K2 = K3.

Table 25 — Defines for TDES Symmetric Cipher Algorithm Constants

Name	Value	Comments
TDES_KEY_SIZES_BITS	{128, 192}	key sizes include the ‘parity’ bit in each byte
TDES_BLOCK_SIZES_BITS	{64, 64}	
TDES_ROUNDS	{48, 48}	DES-equivalent rounds

The following 64, 64-bit DES key values shall not be used in a TCG compliant device.

0101010101010101_{16}	$FEFEFEFEFEFEFEFE_{16}$	$E0E0E0E0F1F1F1F1_{16}$	$1F1F1F1F0E0E0E0E_{16}$
$011F011F010E010E_{16}$	$1F011F010E010E01_{16}$	$01E001E001F101F1_{16}$	$E001E001F101F101_{16}$
$01FE01FE01FE01FE_{16}$	$FE01FE01FE01FE01_{16}$	$1FE01FE00EF10EF1_{16}$	$E01FE01FF10EF10E_{16}$
$1FFE1FFE0EFE0EFE_{16}$	$FE1FFE1FFE0EFE0E_{16}$	$E0FEE0FEF1FEF1FE_{16}$	$FEE0FEE0FEF1FEF1_{16}$
$01011F1F01010E0E_{16}$	$1F1F01010E0E0101_{16}$	$E0E01F1FF1F10E0E_{16}$	$0101E0E00101F1F1_{16}$
$1F1FE0E00E0EF1F1_{16}$	$E0E0FEFEF1F1FEFE_{16}$	$0101FEFE0101FEFE_{16}$	$1F1FFEFE0E0EFEFE_{16}$
$E0FE011FF1FE010E_{16}$	$011F1F01010E0E01_{16}$	$1FE001FE0EF101FE_{16}$	$E0FE1F01F1FE0E01_{16}$
$011FE0FE010EF1FE_{16}$	$1FE0E01F0EF1F10E_{16}$	$E0FEFEE0F1FEFEF1_{16}$	$011FFEE0010EFEF1_{16}$
$1FE0FE010EF1FE01_{16}$	$FE0101FEFE0101FE_{16}$	$01E01FFE01F10EFE_{16}$	$1FFE01E00EFE01F1_{16}$
$FE011FE0FE010EF1_{16}$	$FE01E01FFE01F10E_{16}$	$1FFEE0010EFEF101_{16}$	$FE1F01E0FE0E01F1_{16}$
$01E0E00101F1F101_{16}$	$1FFEFE1F0EFEFE0E_{16}$	$FE1FE001FE0EF101_{16}$	$01E0FE1F01F1FE0E_{16}$
$E00101E0F10101F1_{16}$	$FE1F1FFEFE0E0EFE_{16}$	$01FE1FE001FE0EF1_{16}$	$E0011FFEF1010EFE_{16}$
$FEE0011FFEF1010E_{16}$	$01FEE01F01FEF10E_{16}$	$E001FE1FF101FE0E_{16}$	$FEE01F01FEF10E01_{16}$
$01FEFE0101FEFE01_{16}$	$E01F01FEF10E01FE_{16}$	$FEE0E0FEFEF1F1FE_{16}$	$1F01011F0E01010E_{16}$
$E01F1FE0F10E0EF1_{16}$	$FEFE0101FEFE0101_{16}$	$1F01E0FE0E01F1FE_{16}$	$E01FFE01F10EFE01_{16}$
$FEFE1F1FFEFE0E0E_{16}$	$1F01FEE00E01FEF1_{16}$	$E0E00101F1F10101_{16}$	$FEFEE0E0FEFEF1F1_{16}$

Annex A — Bibliography

For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

- FIPS 202, SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions
- GM/T 0003.1-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 1: General*
- GM/T 0003.2-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 2: Digital Signature Algorithm*
- GM/T 0003.3-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 3: Key Exchange Protocol*
- GM/T 0003.5-2012: *Public Key Cryptographic Algorithm SM2 Based on Elliptic Curves Part 5: Parameter definition*
- GM/T 0004-2012: *SM3 Cryptographic Hash Algorithm*
- GM/T 0002-2012: *SM4 Block Cipher Algorithm*
- IEEE Std 1363™-2000, *Standard Specifications for Public Key Cryptography*
- IEEE Std 1363a™-2004 (Amendment to IEEE Std 1363™-2000), IEEE *Standard Specifications for Public Key Cryptography- Amendment 1: Additional Techniques*
- IETF RFC 8017, Public-Key Cryptography Standards (PKCS) #1: RSA Cryptography Specifications Version 2.2
- ISO/IEC 9797-2, Information technology — Security techniques — Message authentication codes (MACs) — Part 2: Mechanisms using a dedicated hash-function
- ISO/IEC 10116, Information technology — Security techniques — Modes of operation for an *n*-bit block cipher
- ISO/IEC 10118-3, Information technology — Security techniques — Hash-functions — Part 3: Dedicated hash functions
- ISO/IEC 14888-3, Information technology -- Security techniques -- Digital signature with appendix -- Part 3: Discrete logarithm based mechanisms
- ISO/IEC 15946-1, Information technology — Security techniques — Cryptographic techniques based on elliptic curves — Part 1: General
- ISO/IEC 18033-3, Information technology — Security techniques — Encryption algorithms — Part 3: Block ciphers
- NIST SP800-108, Recommendation for Key Derivation Using Pseudorandom Functions (Revised)
- NIST SP800-56A, Recommendation for Pair-Wise Key Establishment Schemes Using Discrete Logarithm Cryptography (Revised)
- TCG Trusted Platform Module 2.0 Library Specification – Part 1: Architecture

