## **XMSS and XMSSMT**

XMSS and XMSSMT mechanisms for single-part signatures and verification, following the digital signature algorithm defined in [RFC 8391] and [NIST 800-208].

*Table 1, XMSS and XMSSMT Mechanisms vs. Functions*

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Mechanism** | **Functions** | | | | | | |
| **Encrypt & Decrypt** | **Sign & Verify** | **SR & VR** | **Digest** | **Gen. Key/ Key Pair** | **Wrap & Unwrap** | **Derive** |
| CKM\_XMSS\_KEY\_PAIR\_GEN |  |  |  |  |  |  |  |
| CKM\_XMSS |  | 1 |  |  |  |  |  |
| CKM\_XMSSMT\_KEY\_PAIR\_GEN |  |  |  |  |  |  |  |
| CKM\_XMSSMT |  | 1 |  |  |  |  |  |

1 Single-part operations only

### Definitions

This section defines the key type **CKK\_XMSS** and **CKK\_XMSSMT** for type **CK\_KEY\_TYPE** as used in the **CKA\_KEY\_TYPE** attribute of key objects and domain parameter objects.

Mechanisms:

CKM\_XMSS\_KEY\_PAIR\_GEN

CKM\_XMSS

CKM\_XMSSMT\_KEY\_PAIR\_GEN

CKM\_XMSSMT

### XMSS public key objects

XMSS public key objects (object class **CKO\_PUBLIC\_KEY,** key type **CKK\_XMSS**) hold XMSS public keys.

The following table defines the XMSS public key object attributes, in addition to the common attributes defined for this object class:

Table 1, XMSS Public Key Object Attributes

| **Attribute** | **Data Type** | **Meaning** |
| --- | --- | --- |
| CKA\_XMSS\_PARAMS1,3 | Byte array | DER-encoding of a Parameters value as defined below. |
| CKA\_VALUE1,4 | Byte array | 28+2\*N bytes; a 4 byte integer in big-endian order encoding an *algorithm OID*, an N byte string (the Merkle tree root hash value), an N byte string (the public SEED). |

Refer to [PKCS11-Base] Table 11 for footnotes

The **CKA\_XMSS\_PARAMS** attribute value is known as the “XMSS domain parameters” and is defined as a choice of one representation method (with the possibility of future alterations by NIST) with the following syntax:

Parameters ::= CHOICE { -- only one possibility for now --

specifiedParams OCTET STRING

}

typedef CK\_ULONG CK\_XMSS\_OID;

typedef struct specifiedParams {

CK\_XMSS\_OID oid;

} specifiedParams;

where the XMSS *oid* is defined in [RFC 8391] and [NIST 800-208]. The following values are valid for **CK\_XMSS\_OID**.

Table 3, CK\_XMSS\_OID values

|  |  |
| --- | --- |
| **Source Identifier** | **Value** |
| CK\_XMSS\_SHA2\_256\_H10\_N32 | 0x00000001UL |
| CK\_XMSS\_SHA2\_256\_H16\_N32 | 0x00000002UL |
| CK\_XMSS\_SHA2\_256\_H20\_N32 | 0x00000003UL |
| CK\_XMSS\_SHA2\_512\_H10\_N641 | 0x00000004UL |
| CK\_XMSS\_SHA2\_512\_H16\_N641 | 0x00000005UL |
| CK\_XMSS\_SHA2\_512\_H20\_N641 | 0x00000006UL |
| CK\_XMSS\_SHAKE\_128\_H10\_N321 | 0x00000007UL |
| CK\_XMSS\_SHAKE\_128\_H16\_N321 | 0x00000008UL |
| CK\_XMSS\_SHAKE\_128\_H20\_N321 | 0x00000009UL |
| CK\_XMSS\_SHAKE\_256\_H10\_N641 | 0x0000000AUL |
| CK\_XMSS\_SHAKE\_256\_H16\_N641 | 0x0000000BUL |
| CK\_XMSS\_SHAKE\_256\_H20\_N641 | 0x0000000CUL |
| CK\_XMSS\_SHA2\_256\_H10\_N24 | TBD |
| CK\_XMSS\_SHA2\_256\_H16\_N24 | TBD |
| CK\_XMSS\_SHA2\_256\_H20\_N24 | TBD |
| CK\_XMSS\_SHAKE\_256\_H10\_N32 | TBD |
| CK\_XMSS\_SHAKE\_256\_H16\_N32 | TBD |
| CK\_XMSS\_SHAKE\_256\_H20\_N32 | TBD |
| CK\_XMSS\_SHAKE\_256\_H10\_N24 | TBD |
| CK\_XMSS\_SHAKE\_256\_H16\_N24 | TBD |
| CK\_XMSS\_SHAKE\_256\_H20\_N24 | TBD |

1 The parameter sets specified in [RFC 8391] that use SHAKE128, SHAKE256, and SHA-512 are not approved by [NIST 800-208]. The value of these parameters may be reused for those marked TBD in future revisions.

The following is a sample template for creating an XMSS public key object:

CK\_OBJECT\_CLASS keyClass = CKO\_PUBLIC\_KEY;

CK\_KEY\_TYPE keyType = CKK\_XMSS;

CK\_UTF8CHAR label[] = “An XMSS public key object”;

CK\_BYTE xmssParams[] = {...};

CK\_BYTE value[] = {...};

CK\_BBOOL true = CK\_TRUE;

CK\_BBOOL false = CK\_FALSE;

CK\_ATTRIBUTE template[] = {

{CKA\_CLASS, &keyClass, sizeof(keyClass)},

{CKA\_KEY\_TYPE, &keyType, sizeof(keyType)},

{CKA\_TOKEN, &false, sizeof(false)},

{CKA\_LABEL, label, sizeof(label)-1},

{CKA\_XMSS\_PARAMS, &xmssParams, sizeof(xmssParams)},

{CKA\_VALUE, value, sizeof(value)},

{CKA\_VERIFY, &true, sizeof(true)}

};

### XMSSMT public key objects

XMSSMT public key objects (object class **CKO\_PUBLIC\_KEY,** key type **CKK\_XMSSMT**) hold XMSSMT public keys. XMSSMT public key object format is as follows:

Parameters ::= CHOICE { -- only one possibility for now --

specifiedParams OCTET STRING

}

typedef CK\_ULONG CK\_XMSSMT\_OID;

typedef struct specifiedParams {

CK\_XMSSMT\_OID oid;

} specifiedParams;

The XMSSMT *oid* is defined in [RFC 8391] and [NIST 800-208]. The following values are valid for CK\_XMSSMT\_OID:

Table 4, CK\_XMSSMT\_TYPE values

|  |  |
| --- | --- |
| **Source Identifier** | **Value** |
| CK\_XMSSMT\_SHA2\_256\_H20\_D2\_N32 | 0x00000001UL |
| CK\_XMSSMT\_SHA2\_256\_H20\_D4\_N32 | 0x00000002UL |
| CK\_XMSSMT\_SHA2\_256\_H40\_D2\_N32 | 0x00000003UL |
| CK\_XMSSMT\_SHA2\_256\_H40\_D4\_N32 | 0x00000004UL |
| CK\_XMSSMT\_SHA2\_256\_H40\_D8\_N32 | 0x00000005UL |
| CK\_XMSSMT\_SHA2\_256\_H60\_D3\_N32 | 0x00000006UL |
| CK\_XMSSMT\_SHA2\_256\_H60\_D6\_N32 | 0x00000007UL |
| CK\_XMSSMT\_SHA2\_256\_H60\_D12\_N32 | 0x00000008UL |
| CK\_XMSSMT\_SHA2\_512\_H20\_D2\_N641 | 0x00000009UL |
| CK\_XMSSMT\_SHA2\_512\_H20\_D4\_N641 | 0x0000000AUL |
| CK\_XMSSMT\_SHA2\_512\_H40\_D2\_N641 | 0x0000000BUL |
| CK\_XMSSMT\_SHA2\_512\_H40\_D4\_N641 | 0x0000000CUL |
| CK\_XMSSMT\_SHA2\_512\_H40\_D8\_N641 | 0x0000000DUL |
| CK\_XMSSMT\_SHA2\_512\_H60\_D3\_N641 | 0x0000000EUL |
| CK\_XMSSMT\_SHA2\_512\_H60\_D6\_N641 | 0x0000000FUL |
| CK\_XMSSMT\_SHA2\_512\_H60\_D12\_N641 | 0x00000010UL |
| CK\_XMSSMT\_SHAKE\_128\_H20\_D2\_N321 | 0x00000011UL |
| CK\_XMSSMT\_SHAKE\_128\_H20\_D4\_N321 | 0x00000012UL |
| CK\_XMSSMT\_SHAKE\_128\_H40\_D2\_N321 | 0x00000013UL |
| CK\_XMSSMT\_SHAKE\_128\_H40\_D4\_N321 | 0x00000014UL |
| CK\_XMSSMT\_SHAKE\_128\_H40\_D8\_N321 | 0x00000015UL |
| CK\_XMSSMT\_SHAKE\_128\_H60\_D3\_N321 | 0x00000016UL |
| CK\_XMSSMT\_SHAKE\_128\_H60\_D6\_N321 | 0x00000017UL |
| CK\_XMSSMT\_SHAKE\_128\_H60\_D12\_N321 | 0x00000018UL |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D2\_N641 | 0x00000019UL |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D4\_N641 | 0x0000001AUL |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D2\_N641 | 0x0000001BUL |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D4\_N641 | 0x0000001CUL |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D8\_N641 | 0x0000001DUL |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D3\_N641 | 0x0000001EUL |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D6\_N641 | 0x0000001FUL |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D12\_N641 | 0x00000020UL |
| CK\_XMSSMT\_SHA2\_256\_H20\_D2\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H20\_D4\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H40\_D2\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H40\_D4\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H40\_D8\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H60\_D3\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H60\_D6\_N24 | TBD |
| CK\_XMSSMT\_SHA2\_256\_H60\_D12\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D2\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D4\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D2\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D4\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D8\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D3\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D6\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D12\_N32 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D2\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H20\_D4\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D2\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D4\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H40\_D8\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D3\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D6\_N24 | TBD |
| CK\_XMSSMT\_SHAKE\_256\_H60\_D12\_N24 | TBD |

1 The parameter sets specified in [RFC 8391] that use SHAKE128, SHAKE256, and SHA-512 are not approved by [NIST 800-208]. The value of these parameters may be reused for those marked TBD in future revisions.

See section 1.1.2 for sample template, using appropriate XMSSMT types.

### XMSS private key objects

XMSS private key objects (object class **CKO\_PRIVATE\_KEY,** key type **CKK\_XMSS**) hold XMSS private keys.

The following table defines the XMSS private key object attributes, in addition to the common attributes defined for this object class:

*Table 5, XMSS Private Key Object Attributes*

| **Attribute** | **Data Type** | **Meaning** |
| --- | --- | --- |
| CKA\_XMSS\_PARAMS1,4,6 | Byte array | DER-encoding of XMSS parameters. |
| CKA\_VALUE1,4,6,7 | Byte array | Vendor defined.  Note that exporting this value is dangerous as it would allow key reuse. |

Refer to [PKCS11-Base] Table 11 for footnotes

The following is a sample template for creating an XMSS private key object:

CK\_OBJECT\_CLASS keyClass = CKO\_PRIVATE\_KEY;

CK\_KEY\_TYPE keyType = CKK\_XMSS;

CK\_UTF8CHAR label[] = “An XMSS private key object”;

CK\_BYTE xmssParams[] = {...};

CK\_BYTE value[] = {...};

CK\_BBOOL true = CK\_TRUE;

CK\_BBOOL false = CK\_FALSE;

CK\_ATTRIBUTE template[] = {

    {CKA\_CLASS, &keyClass, sizeof(keyClass)},

    {CKA\_KEY\_TYPE, &keyType, sizeof(keyType)},

{CKA\_TOKEN, &true, sizeof(true)},

    {CKA\_LABEL, label, sizeof(label)-1},

{CKA\_SENSITIVE, &true, sizeof(true)},

{CKA\_EXTRACTABLE, &false, sizeof(true)},

    {CKA\_XMSS\_PARAMS, xmssParams, sizeof(xmssParams)},

    {CKA\_VALUE, value, sizeof(value)}

{CKA\_SIGN, &true, sizeof(true)}

};

CKA\_SENSITIVE MUST be true and CKA\_EXTRACTABLE MUST be false for this key.

### XMSSMT private key objects

XMSSMT private key objects (object class **CKO\_PRIVATE\_KEY,** key type **CKK\_XMSSMT**) hold XMSSMT private keys.

The following table defines the XMSSMT private key object attributes, in addition to the common attributes defined for this object class:

*Table 5, XMSSMT Private Key Object Attributes*

| **Attribute** | **Data Type** | **Meaning** |
| --- | --- | --- |
| CKA\_XMSSMT\_PARAMS1,4,6 | Byte array | DER-encoding of XMSSMT parameters. |
| CKA\_VALUE1,4,6,7 | Byte array | Vendor defined.  Note that exporting this value is dangerous as it would allow key reuse. |

Refer to [PKCS11-Base] Table 11 for footnotes

See section 1.1.4 for sample template, using appropriate XMSSMT types.

### XMSS key pair generation

The XMSS key pair generation mechanism, denoted **CKM\_XMSS\_KEY\_PAIR\_GEN**, is a key pair generation mechanism for XMSS.

This mechanism does not have a parameter.

The mechanism generates XMSS public/private key pairs using an *oid*, as specified in the **CKA\_XMSS\_PARAMS** attribute of the template for the public key.

The mechanism contributes the **CKA\_CLASS**, **CKA\_KEY\_TYPE**, and **CKA\_VALUE** attributes to the new public key and the **CKA\_CLASS**, **CKA\_KEY\_TYPE**, **CKA\_VALUE**, and **CKA\_XMSS\_PARAMS** attributes to the new private key.

For this mechanism, the *ulMinKeySize* and *ulMaxKeySize* fields of the **CK\_MECHANISM\_INFO** structure are not used and must be set to 0.

### XMSSMT key pair generation

The XMSSMT key pair generation mechanism, denoted **CKM\_XMSSMT\_KEY\_PAIR\_GEN**, is the key pair generation mechanism for XMSSMT.

The mechanism generates XMSS public/private key pairs using an *oid*, as specified in the **CKA\_XMSSMT\_PARAMS** attribute of the template for the public key.

All other restrictions detailed in section 1.1.6 apply, using XMSSMT types where necessary.

### XMSS and XMSSMT without hashing

The XMSS and XMSSMT without hashing mechanisms, denoted **CKM\_XMSS** and **CKM\_XMSSMT** respectively, are mechanisms for single-part signatures and verification.

These mechanisms do not have parameters.

For the purposes of these mechanisms, an XMSS or XMSSMT signature is a byte string with a length depending on the *oid* provided.

Table 6, XMSS without hashing: Key and Data Length

| **Function** | **Key type** | **Input length** | **Output length** |
| --- | --- | --- | --- |
| C\_Sign1 | XMSS Private Key | any | 14922-2820 |
| C\_Verify1 | XMSS Public Key | any, 14922-2820 | N/A |

1 Single-part operations only.  
2 Extrapolated lowest signature size from [NIST 800-208], using *N*=24 and *len*=51.

Table *7*, XMSSMT without hashing: Key and Data Length

| **Function** | **Key type** | **Input length** | **Output length** |
| --- | --- | --- | --- |
| C\_Sign1 | XMSSMT Private Key | any | 29842-27688 |
| C\_Verify1 | XMSSMT Public Key | any, 29842-27688 | N/A |

1 Single-part operations only.  
2 Extrapolated lowest signature size from [NIST 800-208], using *N*=24, *len*=51, and *D*=2.

For these mechanisms, the *ulMinKeySize* and *ulMaxKeySize* fields of the **CK\_MECHANISM\_INFO** structure are not used and must be set to 0.

### XMSS and XMSSMT definitions

**Errors:**  
  
#define CKR\_KEY\_EXHAUSTED 0x00000203UL

**Key types:**  
  
#define CKK\_XMSS 0x00000047UL  
#define CKK\_XMSSMT 0x00000048UL

**Mechanisms:**  
  
#define CKM\_XMSS\_KEY\_PAIR\_GEN 0x00004033UL  
#define CKM\_XMSSMT\_KEY\_PAIR\_GEN 0x00004034UL  
  
#define CKM\_XMSS 0x00004035UL  
#define CKM\_XMSSMT 0x00004036UL

**Attributes:**  
  
#define CKA\_XMSS\_PARAMS 0x00000615UL  
#define CKA\_XMSSMT\_PARAMS 0x00000616UL