

2.1 RSA

Mechanism	Functions						
	Encrypt & Decrypt	Sign & Verify	SR & VR ¹	Digest	Gen. Key/Key Pair	Wrap & Unwrap	Derive
CKM_RSA_PKCS_KEY_PAIR_GEN					✓		
CKM_RSA_X9_31_KEY_PAIR_GEN					✓		
CKM_RSA_PKCS	✓ ²	✓ ²	✓			✓	
CKM_RSA_PKCS_OAEP	✓ ²					✓	
CKM_RSA_PKCS_PSS		✓ ²					
CKM_RSA_9796		✓ ²	✓				
CKM_RSA_X_509	✓ ²	✓ ²	✓			✓	
CKM_RSA_X9_31		✓ ²					
CKM_SHA1_RSA_PKCS		✓					
CKM_SHA256_RSA_PKCS		✓					
CKM_SHA384_RSA_PKCS		✓					
CKM_SHA512_RSA_PKCS		✓					
CKM_SHA1_RSA_PKCS_PSS		✓					
CKM_SHA256_RSA_PKCS_PSS		✓					
CKM_SHA384_RSA_PKCS_PSS		✓					
CKM_SHA512_RSA_PKCS_PSS		✓					
CKM_SHA1_RSA_X9_31		✓					
CKM_RSA_PKCS TPM_1_1	✓ ²					✓	
CKM_RSA_OAEP TPM_1_1	✓ ²					✓	

2.1.1 Definitions

This section defines the RSA key type “CKK_RSA” for type CK_KEY_TYPE as used in the CKA_KEY_TYPE attribute of RSA key objects.

Mechanisms:

- CKM_RSA_PKCS_KEY_PAIR_GEN
- CKM_RSA_PKCS
- CKM_RSA_9796
- CKM_RSA_X_509
- CKM_MD2_RSA_PKCS
- CKM_MD5_RSA_PKCS
- CKM_SHA1_RSA_PKCS
- CKM_SHA224_RSA_PKCS
- CKM_SHA256_RSA_PKCS
- CKM_SHA384_RSA_PKCS
- CKM_SHA512_RSA_PKCS
- CKM_RIPEMD128_RSA_PKCS
- CKM_RIPEMD160_RSA_PKCS
- CKM_RSA_PKCS_OAEP
- CKM_RSA_X9_31_KEY_PAIR_GEN
- CKM_RSA_X9_31
- CKM_SHA1_RSA_X9_31

CKM_RSA_PKCS_PSS
 CKM_SHA1_RSA_PKCS_PSS
 CKM_SHA224_RSA_PKCS_PSS
 CKM_SHA256_RSA_PKCS_PSS
 CKM_SHA512_RSA_PKCS_PSS
 CKM_SHA384_RSA_PKCS_PSS
 CKM_RSA_PKCS TPM_1_1
 CKM_RSA_OAEP TPM_1_1

2.1.2 RSA public key objects

RSA public key objects (object class **CKO_PUBLIC_KEY**, key type **CKK_RSA**) hold RSA public keys. The following table defines the RSA public key object attributes, in addition to the common attributes defined for this object class:

Table 1, RSA Public Key Object Attributes

Attribute	Data type	Meaning
CKA_MODULUS ^{1,4}	Big integer	Modulus n
CKA_MODULUS_BITS ^{2,3}	CK_ULONG	Length in bits of modulus n
CKA_PUBLIC_EXPONENT ¹	Big integer	Public exponent e

¹Refer to [PKCS #11-B] table 15 for footnotes

Depending on the token, there may be limits on the length of key components. See PKCS #1 for more information on RSA keys.

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

```

CK_OBJECT_CLASS class = CKO_PUBLIC_KEY;
CK_KEY_TYPE keyType = CKK_RSA;
CK_UTF8CHAR label[] = "An RSA public key object";
CK_BYTE modulus[] = {...};
CK_BYTE exponent[] = {...};
CK_BBOOL true = CK_TRUE;
CK_ATTRIBUTE template[] = {
  {CKA_CLASS, &class, sizeof(class)},
  {CKA_KEY_TYPE, &keyType, sizeof(keyType)},
  {CKA_TOKEN, &true, sizeof(true)},
  {CKA_LABEL, label, sizeof(label)-1},
  {CKA_WRAP, &true, sizeof(true)},
  {CKA_ENCRYPT, &true, sizeof(true)},
  {CKA_MODULUS, modulus, sizeof(modulus)},
  {CKA_PUBLIC_EXPONENT, exponent, sizeof(exponent)}
};
```

2.1.20 TPM 1.1 PKCS #1 RSA OAEP

The TPM 1.1 PKCS #1 RSA OAEP mechanism, denoted **CKM_RSA_PKCS_OAEP TPM_1_1**, is a multi-purpose mechanism based on the RSA public-key cryptosystem and the OAEP block format defined in PKCS #1, with additional formatting defined in TCG TPM Specification Version 1.2. It supports single-part encryption and decryption; key wrapping; and key unwrapping.

This mechanism does not have a parameter. It differs from the standard PKCS#1 OAEP RSA encryption mechanism in that the plaintext is wrapped in a TPM_BOUND_DATA structure before being submitted to the encryption process and that all of the values of the parameters that are passed to a standard CKM_RSA_PKCS_OAEP operation are fixed. On encryption, the version field of the TPM_BOUND_DATA structure must contain 0x01, 0x01, 0x00, 0x00. On decryption, any structure of the form 0x01, 0x01, 0xXX, 0xYY may be accepted.

This mechanism can wrap and unwrap any secret key of appropriate length. Of course, a particular token may not be able to wrap/unwrap every appropriate-length secret key that it supports. For wrapping, the “input” to the encryption operation is the value of the **CKA_VALUE** attribute of the key that is wrapped; similarly for unwrapping. The mechanism does not wrap the key type or any other information about the key, except the key length; the application must convey these separately. In particular, the mechanism contributes only the **CKA_CLASS** and **CKA_VALUE** (and **CKA_VALUE_LEN**, if the key has it) attributes to the recovered key during unwrapping; other attributes must be specified in the template.

Constraints on key types and the length of the data are summarized in the following table. For encryption and decryption, the input and output data may begin at the same location in memory. In the table, k is the length in bytes of the RSA modulus.

Table 2, PKCS #1 RSA OAEP: Key And Data Length

Function	Key type	Input length	Output length
C_Encrypt ¹	RSA public key	$\leq k-2-40-5$	k
C_Decrypt ¹	RSA private key	k	$\leq k-2-40-5$
C_WrapKey	RSA public key	$\leq k-2-40-5$	k
C_UnwrapKey	RSA private key	k	$\leq k-2-40-5$

¹ Single-part operations only.

For this mechanism, the *ulMinKeySize* and *ulMaxKeySize* fields of the **CK_MECHANISM_INFO** structure specify the supported range of RSA modulus sizes, in bits.

2.1.21 FIPS 186-4

When CKM_RSA_PKCS is operated in FIPS mode, the length of the modulus can only be 1024, 2048, or 3072 bits.