The SunJCE Provider

As described briefly in The SUN Provider, US export regulations at the time restricted the type of cryptographic functionality that could be available in the JDK. A separate API and reference implementation was developed that allowed applications to encrypt/decrypt date. The Java Cryptographic Extension (JCE) was released as a separate "Optional Package" (also briefly known as a "Standard Extension"), and was available for JDK 1.2x and 1.3x. During the development of JDK 1.4, regulations were relaxed enough that JCE (and SunJSSE) could be bundled as part of the JDK.

The following algorithms are available in the SunJCE provider:

The SunJCE Provider Algorithm Names for Engine Classes. List of SunJCE Provider Algorithm Names for Engine Classes

Engine	Algorithm Names
`AlgorithmParameterGenerator`	DiffieHellman
`AlgorithmParameters`	AES Blowfish ChaCha20-Poly1305 DES DESede DiffieHellman GCM OAEP PBE PBES2 PBEWithHmacSHA1AndAES_128 PBEWithHmacSHA224AndAES_128 PBEWithHmacSHA256AndAES_128 PBEWithHmacSHA384AndAES_128 PBEWithHmacSHA512AndAES_128 PBEWithHmacSHA1AndAES_256 PBEWithHmacSHA224AndAES_256 PBEWithHmacSHA256AndAES_256

Engine	Algorithm Names
	PBEWithHmacSHA384AndAES_256 PBEWithHmacSHA512AndAES_256 PBEWithMD5AndDES PBEWithMD5AndTripleDES PBEWithSHA1AndDESede PBEWithSHA1AndRC2_40 PBEWithSHA1AndRC2_128 PBEWithSHA1AndRC4_40 PBEWithSHA1AndRC4_128 RC2
`Cipher`	See The SunJCE Provider Cipher Transformations
`KeyAgreement`	DiffieHellman
`KeyFactory`	DiffieHellman
`KeyGenerator`	AES ARCFOUR Blowfish ChaCha20 DES DESede HmacMD5 HmacSHA1 HmacSHA224 HmacSHA256 HmacSHA384 HmacSHA512 HmacSHA512/224 HmacSHA512/256 HmacSHA3-224

Engine	Algorithm Names
	HmacSHA3-256 HmacSHA3-384 HmacSHA3-512 RC2
KeyPairGenerator	DiffieHellman
KeyStore	JCEKS
Mac	**Draft comment:** JEP 273: DRBG-Based SecureRandom Implementations included HmacSHA512/224, HmacSHA512/256 HmacMD5 HmacSHA1 HmacSHA224 HmacSHA256 HmacSHA384 HmacSHA512 HmacSHA512/224 HmacSHA512/256 HmacSHA3-224 HmacSHA3-256 HmacSHA3-384 HmacSHA3-512 HmacPBESHA1 **Draft comment[from rgallard]:** JDK-8215440 Add several new Mac algorithms to the SunJCE provider HmacPBESHA224 HmacPBESHA256 HmacPBESHA384 HmacPBESHA512 HmacPBESHA512/224 HmacPBESHA512/256

Engine	Algorithm Names
	PBEWithHmacSHA1 PBEWithHmacSHA224 PBEWithHmacSHA256 PBEWithHmacSHA384 PBEWithHmacSHA512
`SecretKeyFactory`	DES DESede PBEWithMD5AndDES PBEWithMD5AndTripleDES PBEWithSHA1AndDESede PBEWithSHA1AndRC2_40 PBEWithSHA1AndRC2_128 PBEWithSHA1AndRC4_40 PBEWithSHA1AndRC4_128 PBKDF2WithHmacSHA1 PBKDF2WithHmacSHA224 PBKDF2WithHmacSHA256 PBKDF2WithHmacSHA384 PBKDF2WithHmacSHA512 PBEWithHmacSHA1AndAES_128 PBEWithHmacSHA224AndAES_128 PBEWithHmacSHA256AndAES_128 PBEWithHmacSHA384AndAES_128 PBEWithHmacSHA512AndAES_128 PBEWithHmacSHA1AndAES_256 PBEWithHmacSHA224AndAES_256 PBEWithHmacSHA256AndAES_256

Engine	Algorithm Names
	PBEWithHmacSHA384AndAES_256 PBEWithHmacSHA512AndAES_256

The following table lists cipher transformations available in the SunJCE provider.

The SunJCE Provider Cipher Transformations. List of SunJCE Provider Cipher Transformations; includes algorithm names, modes, and paddings.

Algorithm Names	Modes	Paddings
AES	ECB, CBC, PCBC, CFB[1], CFB8..CFB128, OFB[Link text cannot be resolved], OFB8..OFB128	NoPadding, PKCS5Padding, ISO10126Padding
AES	**Draft comment:** JDK-8180377 CTR, CTS, GCM	NoPadding
AES_128, AES_192, AES_256	ECB, CBC, OFB, CFB, GCM	NoPadding
AESWrap	ECB	NoPadding
AESWrap_128	ECB	NoPadding
AESWrap_192	ECB	NoPadding
AESWrap_256	ECB	NoPadding
ARCFOUR	ECB	NoPadding
Blowfish, DES, DESede, RC2	ECB, CBC, PCBC, CTR, CTS, CFB[Link text cannot be resolved], CFB8..CFB64, OFB[Link text cannot be resolved], OFB8..OFB64	NoPadding, PKCS5Padding, ISO10126Padding
ChaCha20	None	NoPadding
ChaCha20-Poly1305	None	NoPadding
DESedeWrap	CBC	NoPadding
PBEWithMD5AndDES, PBEWithMD5AndTripleDES[2], PBEWithSHA1AndDESede, PBEWithSHA1AndRC2_40, PBEWithSHA1AndRC2_128, PBEWithSHA1AndRC4_40, PBEWithSHA1AndRC4_128, PBEWithHmacSHA1AndAES_128, PBEWithHmacSHA224AndAES_128, PBEWithHmacSHA256AndAES_128, PBEWithHmacSHA384AndAES_128, PBEWithHmacSHA512AndAES_128, PBEWithHmacSHA1AndAES_256,	CBC	PKCS5Padding

Algorithm Names	Modes	Paddings
PBEWithHmacSHA224AndAES_256, PBEWithHmacSHA256AndAES_256, PBEWithHmacSHA384AndAES_256, PBEWithHmacSHA512AndAES_256		
RSA	ECB	**Draft comment:** JDK-8146293 Add support for RSASSA-PSS Signature algorithm NoPadding, PKCS1Padding, OAEPPadding, OAEPWithMD5AndMGF1Padding, OAEPWithSHA-1AndMGF1Padding, OAEPWithSHA-1AndMGF1Padding, OAEPWithSHA-224AndMGF1Padding, OAEPWithSHA-256AndMGF1Padding, OAEPWithSHA-384AndMGF1Padding, OAEPWithSHA-512AndMGF1Padding, OAEPWithSHA-512/224AndMGF1Padding, OAEPWithSHA-512/2256ndMGF1Padding

Keysize Restrictions

The SunJCE provider uses the following default key sizes (in bits) and enforces the following restrictions:

KeyGenerator

Draft comment: JEP 273: DRBG-Based SecureRandom Implementations included HmacSHA512/224, HmacSHA512/256

The SunJCE Provider Key Size Restrictions. List of SunJCE Provider Key Size Restrictions

Algorithm Name	Default Key size	Restrictions/Comments
AES	128	Key size must be equal to 128, 192, or 256.
ARCFOUR (RC4)	128	Key size must range between 40 and 1024 (inclusive).
Blowfish	128	Key size must be a multiple of 8, ranging from 32 to 448 (inclusive).
ChaCha20	256	Key size must be equal to 256.
DES	56	Key size must be equal to 56.
DESede (Triple DES)	168	Key size must be equal to 112 or 168. A key size of 112 will generate a Triple DES key with 2 intermediate keys, and a key size of 168 will generate a Triple DES key with 3 intermediate keys. Due to the "Meet-In-The-Middle" problem, even though 112 or 168 bits of key material are used, the **effective** key size is 80 or 112 bits respectively.
HmacMD5	512	No key size restriction.
HmacSHA1	512	No key size restriction.
HmacSHA224	224	No key size restriction.
HmacSHA256	256	No key size restriction.
HmacSHA384	384	No key size restriction.

Algorithm Name	Default Key size	Restrictions/Comments
HmacSHA512	512	No key size restriction.
RC2	128	Key size must range between 40 and 1024 (inclusive).

The various Password-Based Encryption (PBE) algorithms use various algorithms to generate key data, and ultimately depends on the targeted Cipher algorithm. For example, PBEWithMD5AndDES will always generate 56-bit keys.

Draft comment:

The SunJCE Provider Key Size Restrictions. ChaCha20-related table rows removed; only applicable for JDK 12.

Algorithm Name	Default Key size	Restrictions/Comments
ChaCha20	256	Key size must be equal to 256.

Draft comment: JDK-8151405

Draft comment: JDK-8173298

KeyPairGenerator. List of KeyPairGenerator algorithms.

Algorithm Name	Default Key size	Restrictions/Comments
Diffie-Hellman (DH)	2048	Key size must be a multiple of 64, ranging from 512 to 1024, plus 1536, 2048, 3072, 4096, 6144, 8192.

Draft comment: JDK-8151405

Draft comment: JDK-8173298

AlgorithmParameterGenerator. List of AlgorithmParameterGenerator algorithms.

Algorithm Name	Default Key size	Restrictions/Comments
Diffie-Hellman (DH)	2048	Key size must be a multiple of 64, ranging from 512 to 1024, plus 2048 and 3072.

---

[1] CFB/OFB with no specified value defaults to the block size of the algorithm. (i.e. AES is 128; Blowfish, DES, DESede, and RC2 are 64.)

[2] PBEWithMD5AndTripleDES is a proprietary algorithm that has not been standardized.