


Version

.NET 8

▾

 Search

System.Security.Cryptography

- > Aes
- > AesCcm
- > AesCng
- > AesCryptoServiceProvider
- > AesGcm
- > AesManaged
- > AsnEncodedData
- > AsnEncodedDataCollection
- > AsnEncodedDataEnumerator
- > AsymmetricAlgorithm
- > AsymmetricKeyExchangeDeformatter
- > AsymmetricKeyExchangeFormatter
- > AsymmetricSignatureDeformatter
- > AsymmetricSignatureFormatter
- > AuthenticationTagMismatchException
- > ChaCha20Poly1305
- CipherMode
- > CngAlgorithm
- > CngAlgorithmGroup
- CngExportPolicies
- > CngKey
- > CngKeyBlobFormat
- CngKeyCreationOptions
- > CngKeyCreationParameters
- CngKeyHandleOpenOptions
- CngKeyOpenOptions
- CngKeyUsages
- > CngProperty
- > CngPropertyCollection
- CngPropertyOptions
- > CngProvider

 **Download PDF**

[Learn](#) / [.NET](#) / [API browser](#) /

# System.Security.Cryptography Namespace

Reference

 [Feedback](#)

In this article

- Classes
- Structs
- Interfaces
- Enums

Provides cryptographic services, including secure encoding and decoding of data, as well as many other operations, such as hashing, random number generation, and message authentication. For more information, see [Cryptographic Services](#).

## Classes

 **Expand table**

<a href="#">Aes</a>	Represents the abstract base class from which all implementations of the Advanced Encryption Standard (AES) must inherit.
<a href="#">AesCcm</a>	Represents an Advanced Encryption Standard (AES) key to be used with the Counter with CBC-MAC (CCM) mode of operation.
<a href="#">AesCng</a>	Provides a Cryptography Next Generation (CNG) implementation of the Advanced Encryption Standard (AES) algorithm.
<a href="#">AesCryptoServiceProvider</a>	Performs symmetric encryption and decryption using the Cryptographic Application Programming Interfaces (CAPI) implementation of the Advanced Encryption Standard (AES) algorithm.
<a href="#">AesGcm</a>	Represents an Advanced Encryption Standard (AES) key to be used with the Galois/Counter Mode (GCM) mode of operation.
<a href="#">AesManaged</a>	Provides a managed implementation of the Advanced Encryption Standard (AES) symmetric algorithm.
<a href="#">AsnEncodedData</a>	Represents Abstract Syntax Notation One (ASN.1)-encoded data.
<a href="#">AsnEncodedDataCollection</a>	Represents a collection of <a href="#">AsnEncodedData</a> objects. This class cannot be inherited.
<a href="#">AsnEncodedDataEnumerator</a>	Provides the ability to navigate through an <a href="#">AsnEncodedDataCollection</a> object. This class cannot be inherited.
<a href="#">AsymmetricAlgorithm</a>	Represents the abstract base class from which all implementations of asymmetric algorithms must inherit.
<a href="#">AsymmetricKeyExchangeDeformatter</a>	Represents the base class from which all asymmetric key exchange deformatters derive.
<a href="#">AsymmetricKeyExchangeFormatter</a>	Represents the base class from which all asymmetric key exchange formatters derive.
<a href="#">AsymmetricSignatureDeformatter</a>	Represents the abstract base class from which all implementations of asymmetric signature deformatters derive.

<a href="#">AsymmetricSignatureFormatter</a>	Represents the base class from which all implementations of asymmetric signature formatters derive.
<a href="#">AuthenticationTagMismatchException</a>	The exception that is thrown when a decryption operation with an authenticated cipher has an authentication tag mismatch.
<a href="#">ChaCha20Poly1305</a>	Represents a symmetric key to be used with the ChaCha20 stream cipher in the combined mode with the Poly1305 authenticator.
<a href="#">CngAlgorithm</a>	Encapsulates the name of an encryption algorithm.
<a href="#">CngAlgorithmGroup</a>	Encapsulates the name of an encryption algorithm group.
<a href="#">CngKey</a>	Defines the core functionality for keys that are used with Cryptography Next Generation (CNG) objects.
<a href="#">CngKeyBlobFormat</a>	Specifies a key BLOB format for use with Microsoft Cryptography Next Generation (CNG) objects.
<a href="#">CngKeyCreationParameters</a>	Contains advanced properties for key creation.
<a href="#">CngPropertyCollection</a>	Provides a strongly typed collection of Cryptography Next Generation (CNG) properties.
<a href="#">CngProvider</a>	Encapsulates the name of a key storage provider (KSP) for use with Cryptography Next Generation (CNG) objects.
<a href="#">CngUIPolicy</a>	Encapsulates optional configuration parameters for the user interface (UI) that Cryptography Next Generation (CNG) displays when you access a protected key.
<a href="#">CryptoConfig</a>	Accesses the cryptography configuration information.
<a href="#">CryptographicAttributeObject</a>	Contains a type and a collection of values associated with that type.
<a href="#">CryptographicAttributeObjectCollection</a>	Contains a set of <a href="#">CryptographicAttributeObject</a> objects.
<a href="#">CryptographicAttributeObjectEnumerator</a>	Provides enumeration functionality for the <a href="#">CryptographicAttributeObjectCollection</a> collection. This class cannot be inherited.
<a href="#">CryptographicException</a>	The exception that is thrown when an error occurs during a cryptographic operation.
<a href="#">CryptographicOperations</a>	Provides methods for use in working with cryptography to reduce the risk of side-channel information leakage.
<a href="#">CryptographicUnexpectedOperationException</a>	The exception that is thrown when an unexpected operation occurs during a cryptographic operation.
<a href="#">CryptoStream</a>	Defines a stream that links data streams to cryptographic transformations.
<a href="#">CspKeyContainerInfo</a>	Provides additional information about a cryptographic key pair. This class cannot be inherited.
<a href="#">CspParameters</a>	Contains parameters that are passed to the cryptographic service provider (CSP) that performs cryptographic computations. This class cannot be inherited.
<a href="#">DeriveBytes</a>	Represents the abstract base class from which all classes that derive byte sequences of a specified length inherit.
<a href="#">DES</a>	Represents the base class for the Data Encryption Standard (DES) algorithm from which all <a href="#">DES</a> implementations must derive.
<a href="#">DESCryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) version of the Data Encryption Standard ( <a href="#">DES</a> ) algorithm. This class cannot be inherited.
<a href="#">DSA</a>	Represents the abstract base class from which all implementations of the Digital Signature Algorithm ( <a href="#">DSA</a> ) must inherit.

<a href="#">DSACng</a>	Provides a Cryptography Next Generation (CNG) implementation of the Digital Signature Algorithm (DSA).
<a href="#">DSACryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the <a href="#">DSA</a> algorithm. This class cannot be inherited.
<a href="#">DSASOpenSsl</a>	Provides an implementation of the Digital Signature Algorithm (DSA) backed by OpenSSL.
<a href="#">DSASignatureDeformatter</a>	Verifies a Digital Signature Algorithm ( <a href="#">DSA</a> ) PKCS#1 v1.5 signature.
<a href="#">DSASignatureFormatter</a>	Creates a Digital Signature Algorithm ( <a href="#">DSA</a> ) signature.
<a href="#">ECAAlgorithm</a>	Represents the abstract class from which elliptic-curve asymmetric algorithms can inherit.
<a href="#">ECCurve.NamedCurves</a>	Represents a factory class for creating named curves.
<a href="#">ECDiffieHellman</a>	Provides an abstract base class that Elliptic Curve Diffie-Hellman (ECDH) algorithm implementations can derive from. This class provides the basic set of operations that all ECDH implementations must support.
<a href="#">ECDiffieHellmanCng</a>	Provides a Cryptography Next Generation (CNG) implementation of the Elliptic Curve Diffie-Hellman (ECDH) algorithm. This class is used to perform cryptographic operations.
<a href="#">ECDiffieHellmanCngPublicKey</a>	Specifies an Elliptic Curve Diffie-Hellman (ECDH) public key for use with the <a href="#">ECDiffieHellmanCng</a> class.
<a href="#">ECDiffieHellmanOpenSsl</a>	Provides an implementation of the Elliptic Curve Diffie-Hellman (ECDH) algorithm backed by OpenSSL.
<a href="#">ECDiffieHellmanPublicKey</a>	Provides an abstract base class from which all <a href="#">ECDiffieHellmanCngPublicKey</a> implementations must inherit.
<a href="#">ECDsa</a>	Provides an abstract base class that encapsulates the Elliptic Curve Digital Signature Algorithm (ECDSA).
<a href="#">ECDsaCng</a>	Provides a Cryptography Next Generation (CNG) implementation of the Elliptic Curve Digital Signature Algorithm (ECDSA).
<a href="#">ECDsaOpenSsl</a>	Provides an implementation of the Elliptic Curve Digital Signature Algorithm (ECDSA) backed by OpenSSL.
<a href="#">FromBase64Transform</a>	Converts a <a href="#">CryptoStream</a> from base 64.
<a href="#">HashAlgorithm</a>	Represents the base class from which all implementations of cryptographic hash algorithms must derive.
<a href="#">HKDF</a>	RFC5869 HMAC-based Extract-and-Expand Key Derivation (HKDF)
<a href="#">HMAC</a>	Represents the abstract class from which all implementations of Hash-based Message Authentication Code (HMAC) must derive.
<a href="#">HMACMD5</a>	Computes a Hash-based Message Authentication Code (HMAC) by using the <a href="#">MD5</a> hash function.
<a href="#">HMACSHA1</a>	Computes a Hash-based Message Authentication Code (HMAC) using the <a href="#">SHA1</a> hash function.
<a href="#">HMACSHA256</a>	Computes a Hash-based Message Authentication Code (HMAC) by using the <a href="#">SHA256</a> hash function.
<a href="#">HMACSHA3_256</a>	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-256 hash function.
<a href="#">HMACSHA3_384</a>	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-384 hash function.
<a href="#">HMACSHA3_512</a>	Computes a Hash-based Message Authentication Code (HMAC) by using the SHA3-512 hash function.
<a href="#">HMACSHA384</a>	Computes a Hash-based Message Authentication Code (HMAC) using the <a href="#">SHA384</a> hash function.

<a href="#">HMACSHA512</a>	Computes a Hash-based Message Authentication Code (HMAC) using the <a href="#">SHA512</a> hash function.
<a href="#">IncrementalHash</a>	Provides support for computing a hash or HMAC value incrementally across several segments.
<a href="#">KeyedHashAlgorithm</a>	Represents the abstract class from which all implementations of keyed hash algorithms must derive.
<a href="#">KeySizes</a>	Determines the set of valid key sizes for the symmetric cryptographic algorithms.
<a href="#">MaskGenerationMethod</a>	Represents the abstract class from which all mask generator algorithms must derive.
<a href="#">MD5</a>	Represents the abstract class from which all implementations of the <a href="#">MD5</a> hash algorithm inherit.
<a href="#">MD5CryptoServiceProvider</a>	Computes the <a href="#">MD5</a> hash value for the input data using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
<a href="#">Oid</a>	Represents a cryptographic object identifier. This class cannot be inherited.
<a href="#">OidCollection</a>	Represents a collection of <a href="#">Oid</a> objects. This class cannot be inherited.
<a href="#">OidEnumerator</a>	Provides the ability to navigate through an <a href="#">OidCollection</a> object. This class cannot be inherited.
<a href="#">PasswordDeriveBytes</a>	Derives a key from a password using an extension of the PBKDF1 algorithm.
<a href="#">PbeParameters</a>	Represents parameters to be used for Password-Based Encryption (PBE).
<a href="#">PemEncoding</a>	Provides methods for reading and writing the IETF RFC 7468 subset of PEM (Privacy-Enhanced Mail) textual encodings. This class cannot be inherited.
<a href="#">PKCS1MaskGenerationMethod</a>	Computes masks according to PKCS #1 for use by key exchange algorithms.
<a href="#">ProtectedData</a>	Provides methods for encrypting and decrypting data. This class cannot be inherited.
<a href="#">RandomNumberGenerator</a>	Provides functionality for generating random values.
<a href="#">RC2</a>	Represents the base class from which all implementations of the <a href="#">RC2</a> algorithm must derive.
<a href="#">RC2CryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the <a href="#">RC2</a> algorithm. This class cannot be inherited.
<a href="#">Rfc2898DeriveBytes</a>	Implements password-based key derivation functionality, PBKDF2, by using a pseudo-random number generator based on <a href="#">HMACSHA1</a> .
<a href="#">Rijndael</a>	Represents the base class from which all implementations of the <a href="#">Rijndael</a> symmetric encryption algorithm must inherit.
<a href="#">RijndaelManaged</a>	Accesses the managed version of the <a href="#">Rijndael</a> algorithm. This class cannot be inherited.
<a href="#">RNGCryptoServiceProvider</a>	Implements a cryptographic Random Number Generator (RNG) using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
<a href="#">RSA</a>	Represents the base class from which all implementations of the <a href="#">RSA</a> algorithm inherit.
<a href="#">RSACng</a>	Provides a Cryptography Next Generation (CNG) implementation of the RSA algorithm.
<a href="#">RSACryptoServiceProvider</a>	Performs asymmetric encryption and decryption using the implementation of the <a href="#">RSA</a> algorithm provided by the cryptographic

	service provider (CSP). This class cannot be inherited.
<a href="#">RSAEncryptionPadding</a>	Specifies the padding mode and parameters to use with RSA encryption or decryption operations.
<a href="#">RSAOAEPKeyExchangeDeformatter</a>	Decrypts Optimal Asymmetric Encryption Padding (OAEP) key exchange data.
<a href="#">RSAOAEPKeyExchangeFormatter</a>	Creates Optimal Asymmetric Encryption Padding (OAEP) key exchange data using <a href="#">RSA</a> .
<a href="#">RSASOpenSsl</a>	Provides an implementation of the RSA algorithm backed by OpenSSL.
<a href="#">RSAPKCS1KeyExchangeDeformatter</a>	Decrypts the PKCS #1 key exchange data.
<a href="#">RSAPKCS1KeyExchangeFormatter</a>	Creates the PKCS#1 key exchange data using <a href="#">RSA</a> .
<a href="#">RSAPKCS1SignatureDeformatter</a>	Verifies an <a href="#">RSA</a> PKCS #1 version 1.5 signature.
<a href="#">RSAPKCS1SignatureFormatter</a>	Creates an <a href="#">RSA</a> PKCS #1 version 1.5 signature.
<a href="#">RSASignaturePadding</a>	Specifies the padding mode and parameters to use with RSA signature creation or verification operations.
<a href="#">SafeEvPKeyHandle</a>	Represents the <code>EVP_PKEY*</code> pointer type from OpenSSL.
<a href="#">SHA1</a>	Computes the <a href="#">SHA1</a> hash for the input data.
<a href="#">SHA1CryptoServiceProvider</a>	Computes the <a href="#">SHA1</a> hash value for the input data using the implementation provided by the cryptographic service provider (CSP). This class cannot be inherited.
<a href="#">SHA1Managed</a>	Computes the <a href="#">SHA1</a> hash for the input data using the managed library.
<a href="#">SHA256</a>	Computes the <a href="#">SHA256</a> hash for the input data.
<a href="#">SHA256CryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the <a href="#">SHA256</a> algorithm.
<a href="#">SHA256Managed</a>	Computes the <a href="#">SHA256</a> hash for the input data using the managed library.
<a href="#">SHA3_256</a>	Computes the SHA3-256 hash for the input data.
<a href="#">SHA3_384</a>	Computes the SHA3-384 hash for the input data.
<a href="#">SHA3_512</a>	Computes the SHA3-512 hash for the input data.
<a href="#">SHA384</a>	Computes the <a href="#">SHA384</a> hash for the input data.
<a href="#">SHA384CryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the <a href="#">SHA384</a> algorithm.
<a href="#">SHA384Managed</a>	Computes the <a href="#">SHA384</a> hash for the input data using the managed library.
<a href="#">SHA512</a>	Computes the <a href="#">SHA512</a> hash for the input data.
<a href="#">SHA512CryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) implementation of the <a href="#">SHA512</a> algorithm.
<a href="#">SHA512Managed</a>	Computes the <a href="#">SHA512</a> hash algorithm for the input data using the managed library.
<a href="#">Shake128</a>	Computes the SHAKE128 hash for the input data.
<a href="#">Shake256</a>	Computes the SHAKE256 hash for the input data.
<a href="#">SignatureDescription</a>	Contains information about the properties of a digital signature.
<a href="#">SP800108HmacCounterKdf</a>	NIST SP 800-108 HMAC CTR Key-Based Key Derivation (KBKDF)


<a href="#">SymmetricAlgorithm</a>	Represents the abstract base class from which all implementations of symmetric algorithms must inherit.
<a href="#">ToBase64Transform</a>	Converts a <a href="#">CryptoStream</a> to base 64.
<a href="#">TripleDES</a>	Represents the base class for Triple Data Encryption Standard algorithms from which all <a href="#">TripleDES</a> implementations must derive.
<a href="#">TripleDESCng</a>	Provides a Cryptography Next Generation (CNG) implementation of the Triple Data Encryption Standard (3DES) algorithm.
<a href="#">TripleDESCryptoServiceProvider</a>	Defines a wrapper object to access the cryptographic service provider (CSP) version of the <a href="#">TripleDES</a> algorithm. This class cannot be inherited.

## Structs

 Expand table


<a href="#">CngProperty</a>	Encapsulates a property of a Cryptography Next Generation (CNG) key or provider.
<a href="#">DSAParameters</a>	Contains the typical parameters for the <a href="#">DSA</a> algorithm.
<a href="#">ECCurve</a>	Represents an elliptic curve.
<a href="#">ECPParameters</a>	Represents the standard parameters for the elliptic curve cryptography (ECC) algorithm.
<a href="#">ECPPoint</a>	Represents a (X,Y) coordinate pair for elliptic curve cryptography (ECC) structures.
<a href="#">HashAlgorithmName</a>	Specifies the name of a cryptographic hash algorithm.
<a href="#">PemFields</a>	Contains information about the location of PEM data.
<a href="#">RSAParameters</a>	Represents the standard parameters for the <a href="#">RSA</a> algorithm.

## Interfaces

 Expand table


<a href="#">ICryptoTransform</a>	Defines the basic operations of cryptographic transformations.
<a href="#">ICspAsymmetricAlgorithm</a>	Defines methods that allow an <a href="#">AsymmetricAlgorithm</a> class to enumerate key container information, and import and export Microsoft Cryptographic API (CAPI)-compatible key blobs.

## Enums


 Expand table

<a href="#">CipherMode</a>	Specifies the block cipher mode to use for encryption.
<a href="#">CngExportPolicies</a>	Specifies the key export policies for a key.
<a href="#">CngKeyCreationOptions</a>	Specifies options used for key creation.
<a href="#">CngKeyHandleOpenOptions</a>	Specifies options for opening key handles.
<a href="#">CngKeyOpenOptions</a>	Specifies options for opening a key.
<a href="#">CngKeyUsages</a>	Specifies the cryptographic operations that a Cryptography Next Generation (CNG) key may be used with.
<a href="#">CngPropertyOptions</a>	Specifies Cryptography Next Generation (CNG) key property options.


<a href="#">CngUIProtectionLevels</a>	Specifies the protection level for the key in user interface (UI) prompting scenarios.
<a href="#">CryptoStreamMode</a>	Specifies the mode of a cryptographic stream.
<a href="#">CspProviderFlags</a>	Specifies flags that modify the behavior of the cryptographic service providers (CSP).
<a href="#">DataProtectionScope</a>	Specifies the scope of the data protection to be applied by the <a href="#">Protect(Byte[], Byte[], DataProtectionScope)</a> method.
<a href="#">DSASignatureFormat</a>	Specifies the data format for signatures with the DSA family of algorithms.
<a href="#">ECCurve.ECCurveType</a>	Indicates how to interpret the data contained in an <a href="#">ECCurve</a> object.
<a href="#">ECDiffieHellmanKeyDerivationFunction</a>	Specifies the key derivation function that the <a href="#">ECDiffieHellmanCng</a> class will use to convert secret agreements into key material.
<a href="#">EKeyXmlFormat</a>	Defines XML serialization formats for elliptic curve keys.
<a href="#">FromBase64TransformMode</a>	Specifies whether white space should be ignored in the base 64 transformation.
<a href="#">KeyNumber</a>	Specifies whether to create an asymmetric signature key or an asymmetric exchange key.
<a href="#">OidGroup</a>	Identifies Windows cryptographic object identifier (OID) groups.
<a href="#">PaddingMode</a>	Specifies the type of padding to apply when the message data block is shorter than the full number of bytes needed for a cryptographic operation.
<a href="#">PbeEncryptionAlgorithm</a>	Specifies encryption algorithms to be used with Password-Based Encryption (PBE).
<a href="#">RSAEncryptionPaddingMode</a>	Specifies the padding mode to use with RSA encryption or decryption operations.
<a href="#">RSASignaturePaddingMode</a>	Specifies the padding mode to use with RSA signature creation or verification operations.


 **Collaborate with us on GitHub**

The source for this content can be found on GitHub, where you can also create and review issues and pull requests. For more information, see [our contributor guide](#).

 **.NET feedback**

.NET is an open source project. Select a link to provide feedback:

 [Open a documentation issue](#)

 [Provide product feedback](#)