Session: 16

Encrypting and Decrypting Data

Objectives

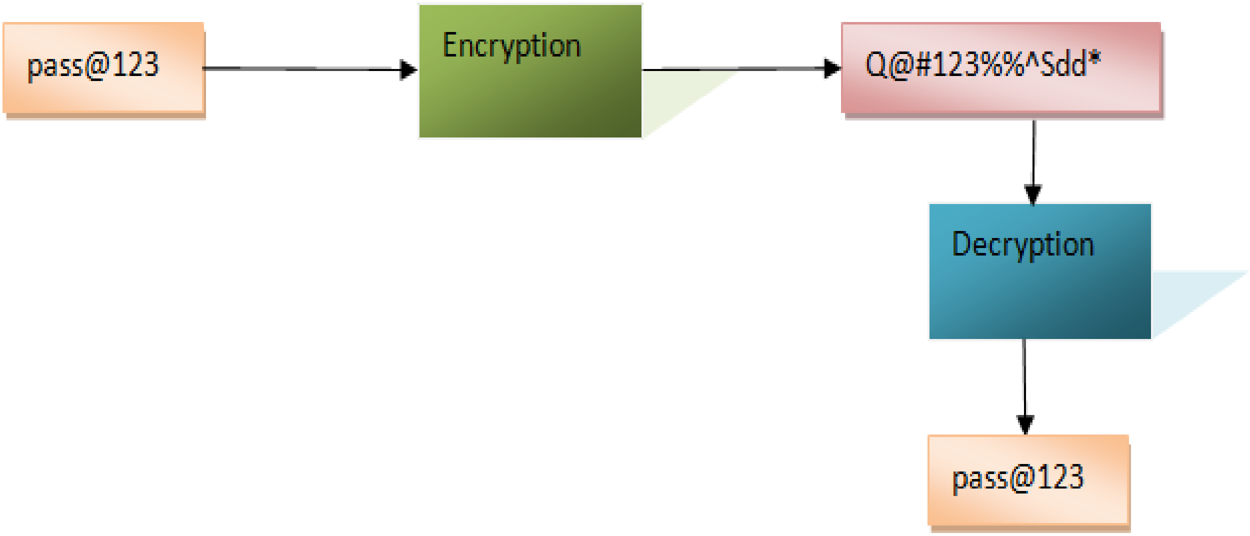
- Explain symmetric encryption.
- Explain asymmetric encryption.
- List the various types in the **System.Security.Cryptography** namespace that supports symmetric and asymmetric encryptions.

Introducing Cryptography and Encryption

- All organizations need to handle sensitive data that can be either present in storage or may be exchanged between different entities within and outside the organization over a network.
- Such data is often prone to misuse either intentionally with malicious intent or unintentionally.
- To avoid such misuse, there should be some security mechanism that can ensure confidentiality and integrity of data.

Encryption Basics

- The following figure shows the process of encryption and decryption of a password as an example.

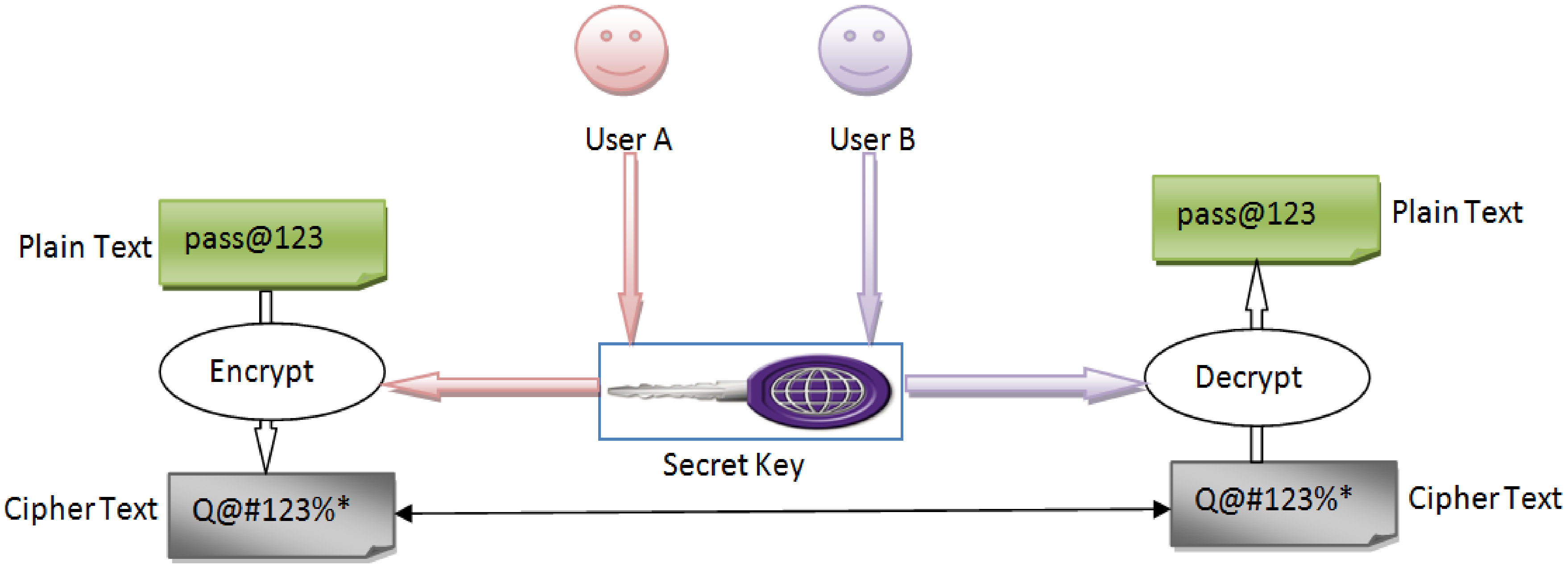


- In the figure:
  - The plain text,pass@123 is encrypted to a cipher text.
  - The cipher text is decrypted back to the original plain text.

Symmetric Encryption

- Symmetric encryption, or secret key encryption, uses a single key, known as the secret key both to encrypt and decrypt data.

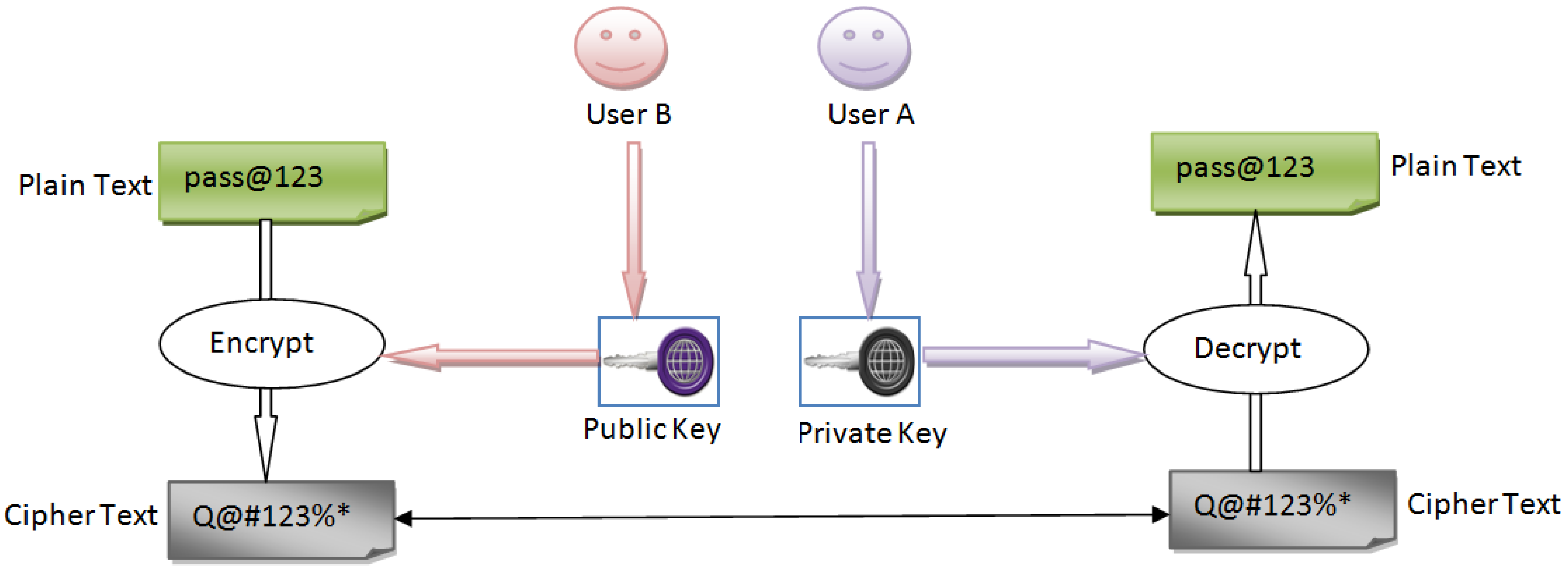
- User A uses a secret key to encrypt a plain text to cipher text.
- User A shares the cipher text and the secret key with User B.
- User B uses the secret key to decrypt the cipher text back to the original plain text.



Asymmetric Encryption

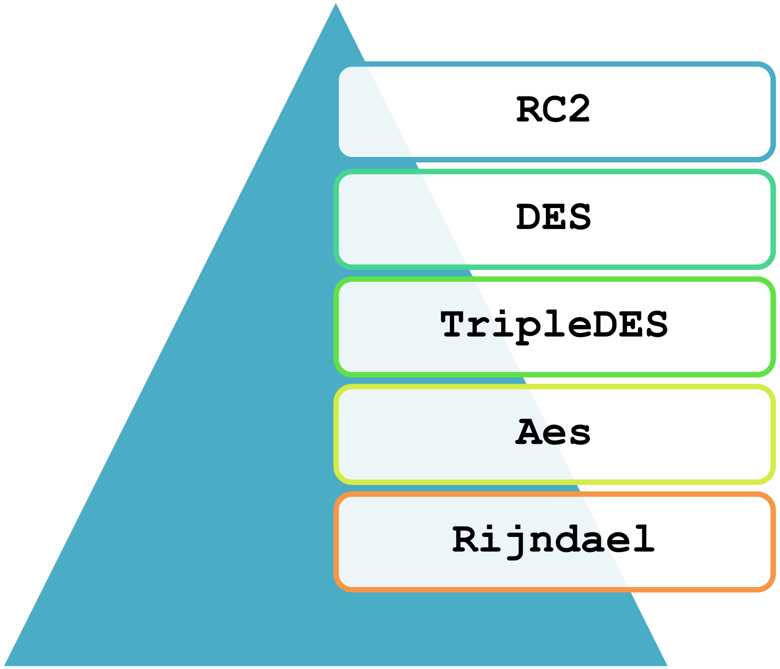
- Asymmetric encryption uses a pair of public and private key to encrypt and decrypt data.

- User A generates a public and private key pair.
- User A shares the public key with User B.
- User B uses the public key to encrypt a plain text to cipher text.
- User A uses the private key to decrypt the cipher text back to the original plain text.



Symmetric Encryption Algorithms 1-6

- The **System.Security.Cryptography** namespace provides the **SymmetricAlgorithm** base class for all symmetric algorithm classes.
- The derived classes of the **SymmetricAlgorithm** base class are as follows:



```
RC2
DES
TripleDES
Aes
Rijndael
```

Symmetric Encryption Algorithms 2-6

RC2

- Is an abstract base class for all classes that implement the RC2 algorithm.
- Is a proprietary algorithm developed by RSA Data Security, Inc. in 1987.
- Supports key sizes ranging from **40 bits to 128** bits in 8-bit increments for encryption.
- Was designed for the old generation processors and currently have been replaced by more faster and secure algorithms.
- Derives the **RC2CryptoServiceProvider** class to provide an implementation of the RC2 algorithm.

Symmetric Encryption Algorithms 3-6

DES

- Is an abstract base class for all classes that implement the Data Encryption Standard (DES) algorithm.
- Developed by IBM but as of today available as a U.S. Government Federal Information Processing Standard (FIPS 46-3).
- Works on the data to encrypt as blocks where each block is of 64 bits.
- Uses a **key of 64** bits to perform the encryption. On account of its small key size DES encrypts data faster as compared to other symmetric algorithms.
- Is prone to brute force security attacks because of its smaller key size.
- Derives the **DESCryptoServicerProvider** class to provide an implementation of the DES algorithm.

Symmetric Encryption Algorithms 4-6

TripleDES

- Is an abstract base class for all the classes that implement the TripleDES algorithm.
- Is an enhancement to the DES algorithm for the purpose of making the DES algorithm more secure against security threats.
- Works on 64 bit blocks that is similar to the DES algorithm.
- Supports key sizes of **128 bits to 192 bits.**
- Derives the **TripleDESCryptoServiceProvider** class to provide an implementation of the TripleDES algorithm.

Symmetric Encryption Algorithms 5-6

Aes

- Is an abstract base class for all classes that implement the Advanced Encryption Standard (AES) algorithm.
- Is a successor of DES and is currently considered as one of the most secure algorithm.
- Is more efficient in encrypting large volume of data in the size of several gigabytes.
- Works on 128-bits blocks of data and key sizes of **128, 192, or 256** bits for encryption.
- Derives **the AesCryptoServiceProvider** and **AesManaged** classes to provide an implementation of the AES algorithm.

Symmetric Encryption Algorithms 6-6

Rijndael

- Is an abstract base class for all the classes that implement the Rijndael algorithm.
- Is a superset of the Aes algorithm.
- Supports key sizes of **128, 192, or 256** bits similar to the Aes algorithm.
- Can have block sizes of 128, 192, or 256 bits, unlike Aes, which has a fixed block size of 128 bits.
- Provides the flexibility to select an appropriate block size based on the volume of data to encrypt by supporting different block sizes.
- Derives the **RijndealManaged** class to provide an implementation of the Rijndeal algorithm.

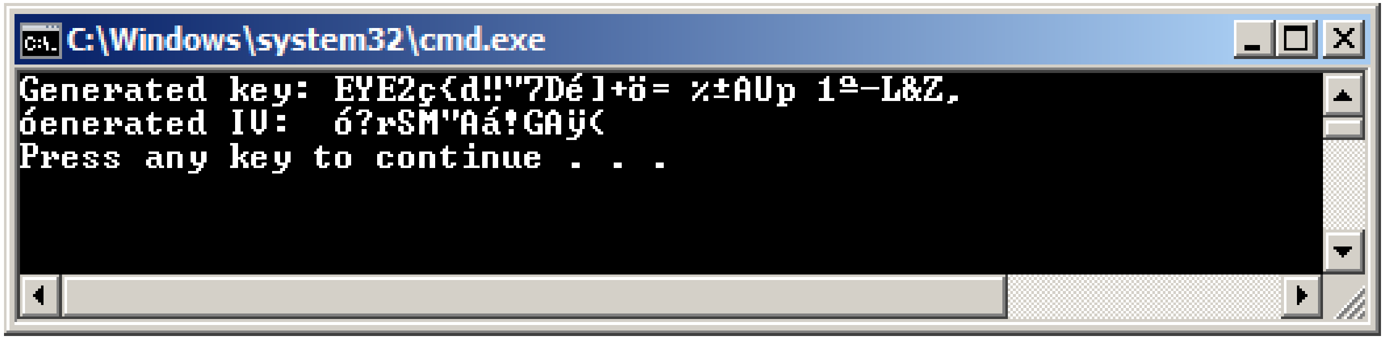
Performing Symmetric Encryption

- Developers can use the derived classes of the SymmetricAlgorithm base class in the namespace **System.Security.Cryptography** to perform symmetric encryption.
- This algorithm functions are classified in three steps:
  - key generation
  - encryption
  - decryption

Generating Symmetric Keys

- When using the default constructor of the symmetric encryption classes, such as RijndaelManaged and AesManaged, a key and IV are automatically generated.
- The generated key and the IV can be accessed as byte arrays using the Key and IV properties of the encryption class.
- Generating Symmetric Keys

```
RijndaelManaged symAlgo = new RijndaelManaged();
Console.WriteLine("Generated key: {0}, \nGenerated IV:  {1}",
Encoding.Default.GetString(symAlgo.Key),
Encoding.Default.GetString(symAlgo.IV));
```



```
Generated key: EYE2ç{d‼"7Dé]+ö= %±AUp 1º-L&Z,
óenerated IV:  ó?rSM"Aá!GAÿ(
Press any key to continue . . .
```

Encrypting Data

- The symmetric encryption classes of the .NET Framework provides the CreateEncryptor() method that returns an object of the ICryptoTransform interface.

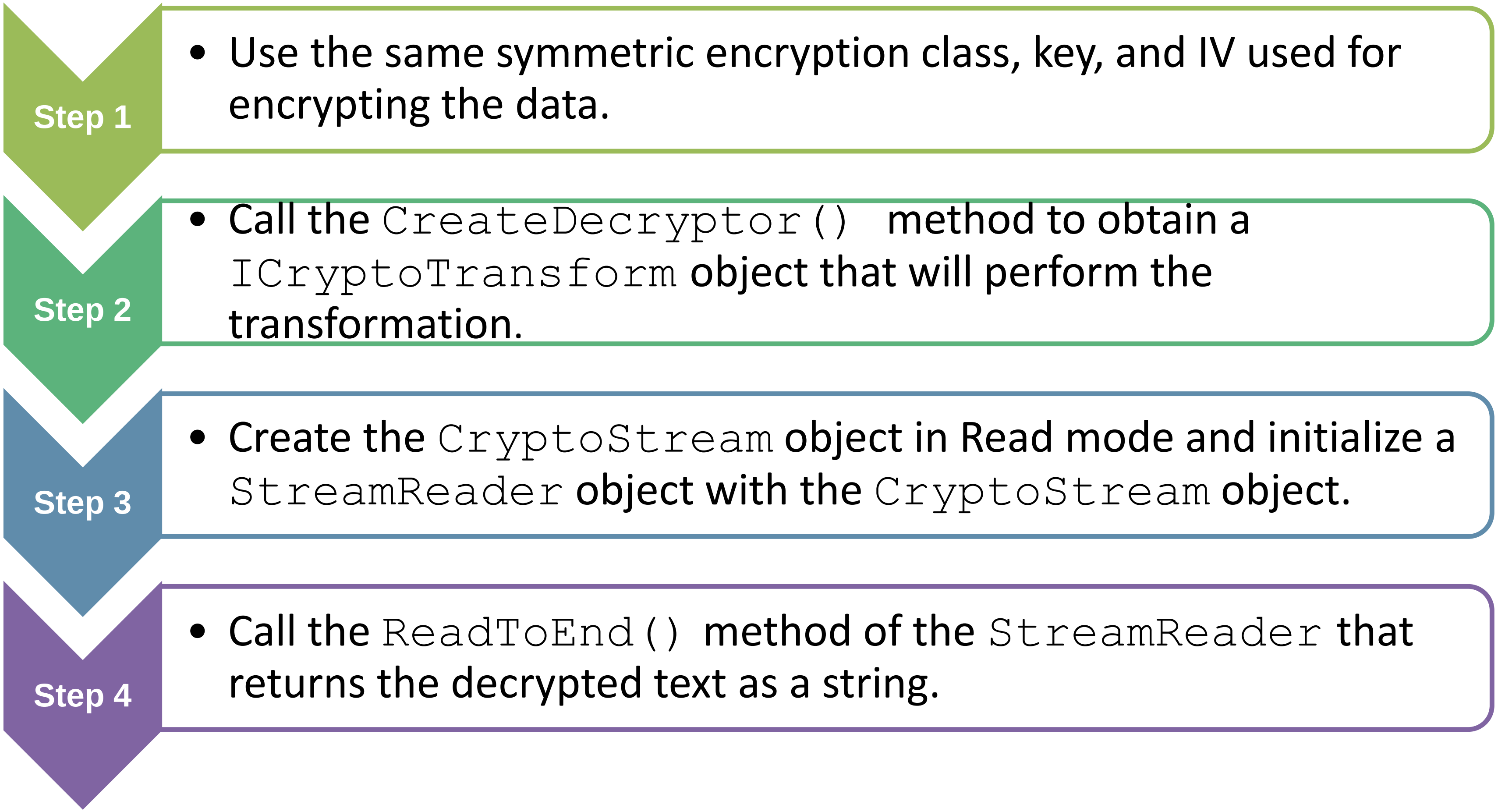
- The ICryptoTransform object is responsible for transforming the data based on the algorithm of the encryption class.

- Once you have obtained an ICryptoTransform object, can use the CryptoStream class to perform encryption.

- The CryptoStream class acts as a wrapper of a stream-derived class, such as FileStream, MemoryStream, and NetworkStream.

Decrypting Data

- To decrypt data, you need to:



Step 1

- Use the same symmetric encryption class, key, and IV used for encrypting the data.

Step 2

- Call the `CreateDecryptor()` method to obtain a `ICryptoTransform` object that will perform the transformation.

Step 3

- Create the `CryptoStream` object in Read mode and initialize a `StreamReader` object with the `CryptoStream` object.

Step 4

- Call the `ReadToEnd()` method of the `StreamReader` that returns the decrypted text as a string.

Asymmetric Encryption Algorithm

- The **System.Security.Cryptography** namespace provides the **AsymmetricAlgorithm** base class for all asymmetric algorithm classes.

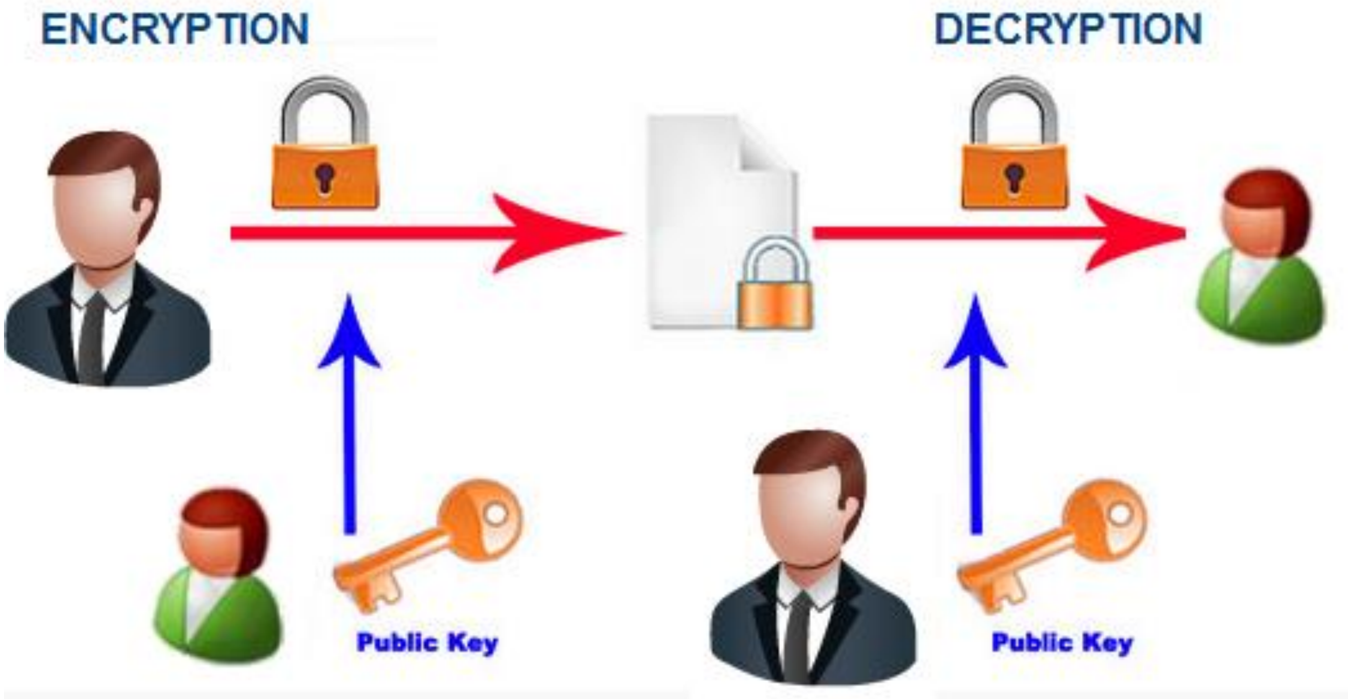
- **RSA** is an abstract class that derives from the **AsymmetricAlgorithm** class.

- The **RSA** class is the base class for all the classes that implement the RSA algorithm.

- The RSA algorithm was designed in 1977 by Ron **R**ivest, Adi **S**hamir, and Leonard **A**dleman and till now is the most widely adopted algorithm to perform asymmetric encryption and decryption.

Performing Asymmetric Encryption

- You can use the RSACryptoServiceProvider class to perform asymmetric encryption.



Encrypting & Decrypting Data

- To encrypt data, you need to:
  - Create a new instance of the **`RSACryptoServiceProvider`** class
  - Call the `ImportParameters()` method to initialize the instance with the public key information exported to an `RSAParameters` structure.
- To decrypt data, you need to initialize an `RSACryptoServiceProvider` object using the private key of the key pair whose public key was used for encryption.

```
RSACryptoServiceProviderRSAKeyGenerator = new
RSACryptoServiceProvider();
RSAParametersrSAKeyInfo = rSAKeyGenerator.ExportParameters(true);
RSACryptoServiceProviderrsaDecryptor= new
RSACryptoServiceProvider();
rsaDecryptor.ImportParameters(rSAKeyInfo);
```

Summary

- Encryption is a security mechanism that converts data in plain text to cipher text.
- An encryption key is a piece of information or parameter that determines how a particular encryption mechanism works.
- The .NET Framework provides various types in the `System.Security.Cryptography` namespace to support symmetric and asymmetric encryptions.
- When you use the default constructor to create an object of the symmetric encryption classes, a key and IV are automatically generated.
- The `ICryptoTransform` object is responsible for transforming the data based on the algorithm of the encryption class.
- The `CryptoStream` class acts as a wrapper of a stream-derived class, such as `FileStream`, `MemoryStream`, and `NetworkStream`.
- When you call the default constructor of the `RSACryptoServiceProvider` and `DSACryptoServiceProvider` classes, a new public/private key pair is automatically generated.