

Encrypting data.

```
using System;
using System.IO;
using System.Security;
using System.Security.Cryptography;

public class StoreCryptoStream : ICryptoStream {
    static byte[] tag1 = {(byte)'[',(byte)'S',(byte)'a',(byte)'u'
    ,
    (byte)'d' ,(byte)'e',(byte)'s' ,(byte)']'};
    static byte[] tag2 = {(byte)'[',(byte)'S',(byte)'a',(byte)'u'
    ,
    (byte)'r' ,(byte)'c',(byte)'2' ,(byte)']'};

    FileStream fs;

    public StoreCryptoStream(FileStream fout) {
        fs = fout;
    }
    public virtual void CloseStream() { fs.Close(); }
    public virtual void CloseStream(Object obj) { fs.Close(); }
    public virtual void SetSink(ICryptoStream pstm) { }
    public virtual void SetSource(CryptographicObject co) { }
    public virtual ICryptoStream GetSink() { return null; }

    public virtual void Write(byte[] bin) {
        int len = bin.GetLength(0);
        Write(bin, 0, len);
    }

    public virtual void Write(byte[] bin, int start, int len) {
        fs.Write(bin, start, len);
    }
}

public class MainClass {
    static byte[] symKey;
```

```

static byte[] symIV;

private static bool GenerateKey(string password) {
try {

int len;
char[] cp = password.ToCharArray();
len = cp.GetLength(0);
byte[] bt = new byte[len];

for (int i = 0; i < len; i++) { bt[i] = (byte)cp[i]; } symKey
= new byte[8]; symIV = new byte[8]; SHA1_CSP sha = new
SHA1_CSP(); sha.Write(bt); sha.CloseStream(); for (int i = 0;
i < 8; i++) { symKey[i] = sha.Hash[i]; } for (int i = 8; i <
16; i++) { symIV[i - 8] = sha.Hash[i]; } return true; } catch
(Exception e) { Console.WriteLine("An Exception Occurred in
Generating eys:" + e.ToString()); return false; } } private
static void EncryptData(string infile, string outfile) { try {
FileStream fin = new FileStream(infile, FileMode.Open,
FileAccess.Read); FileStream fout = new FileStream(outfile,
FileMode.OpenOrCreate, FileAccess.Write); fout.SetLength(0);
byte[] bin = new byte[4096]; long totlen = fin.Length; long
rdlen = 0; int len; SymmetricAlgorithm des = new DES_CSP();
StoreCryptoStream scs = new StoreCryptoStream(fout);
SymmetricStreamEncryptor sse = des.CreateEncryptor(symKey,
symIV); SHA1_CSP sha = new SHA1_CSP(); sse.SetSink(sha);
sha.SetSource(sse); sha.SetSink(scs); scs.SetSource(sha);
while (rdlen < totlen) { len = fin.Read(bin, 0, 4096);
sse.Write(bin, 0, len); rdlen = rdlen + len; }
sse.CloseStream(); fin.Close(); fout.Close(); } catch
(Exception e) { Console.WriteLine("An exception occurred while
encrypting :" + e.ToString()); } } public static void
Main(string[] args) { GenerateKey(args[0]);
EncryptData(args[1], args[2]); } } [/csharp]

```