

## Forum:[Hack Forums](#)

Thread: [\[VB.NET & C#\] PolyAES - Very high security AES class - Unique output - 256-bit - Salt](#)

hgdfhsdgdgdfh - July 16, 2011, 10:12 am

Forget using static and UNSECURE IV's and unsecure algorithms.

Features of this AES class:

- Uses always 256-bit AES
- As secure as possible (uses standardized algorithms)
- "Polymorphic" / unique output (CBC cipher mode with random IV)
- Simple to use
- Works with .NET 2.0

Why its secure:

First, it uses AES, algorithm approved by US government to encrypt TOP SECRET information.

Secondly, it uses standard PBKDF2 algorithm to derive encryption key from password key with RANDOM 256-bit salt and RANDOM IV (= dictionary attacks practically impossible).

Salt and IV are automatically stored concatenated after the actual ciphertext (its not a security risk). You dont have to care about them. Decryption function automatically strips them and uses them in encryption. New completely random salt and IV is generated for each crypt automatically.

REMEMBER TO USE ENOUGH BIG PASSWORD/KEY (I SUGGEST AT LEAST 256-BITS = 32 BYTES!)

VB.NET

Code '

\* Copyright (C) 2011, Dextrey (0xDEADDEAD)

\* Removing this copyright notice is prohibited without permission from author

\* Using this code in your own software product, commercial or not is allowed

,

Imports System.Collections.Generic

Imports System.Text

Namespace DexCrypt

Class PolyAES

Const saltSize As Integer = 32

Private algo As System.Security.Cryptography.SymmetricAlgorithm

Private rngAlgo As System.Security.Cryptography.RNGCryptoServiceProvider

Private salt As Byte()

Public Sub New()

Me.algo = New System.Security.Cryptography.RijndaelManaged()

Me.algo.Mode = System.Security.Cryptography.CipherMode.CBC

Me.rngAlgo = New System.Security.Cryptography.RNGCryptoServiceProvider()

End Sub

Private Sub InitializeSecureParameters(key As Byte())

    ' init rijndael IV

    Me.algo.GenerateIV()

    salt = New Byte(saltSize - 1) {}

    rngAlgo.GetBytes(salt)

2000)    Dim pwDeriveAlg As New System.Security.Cryptography.Rfc2898DeriveBytes(key, salt,

        Me.algo.Key = pwDeriveAlg.GetBytes(32)

End Sub

Private Sub LoadSecureParameters(key As Byte(), encIv As Byte(), encSalt As Byte())

    Me.algo.IV = encIv

    Me.salt = encSalt

2000)    Dim pwDeriveAlg As New System.Security.Cryptography.Rfc2898DeriveBytes(key, salt,

        Me.algo.Key = pwDeriveAlg.GetBytes(32)

End Sub

Public Function Encrypt(plainText As String, key As String) As String

    Return Convert.ToBase64String(Me.Encrypt(UnicodeEncoding.UTF8.GetBytes(plainText),  
UnicodeEncoding.UTF8.GetBytes(key)))

End Function

Public Function Decrypt(cipherText As String, key As String) As String

    Return

UnicodeEncoding.UTF8.GetString(Me.Decrypt(Convert.FromBase64String(cipherText),  
UnicodeEncoding.UTF8.GetBytes(key)))

End Function

Public Function Encrypt(plainText As Byte(), key As Byte()) As Byte()

    InitializeSecureParameters(key)

    Dim encTransform As System.Security.Cryptography.ICryptoTransform =  
algo.CreateEncryptor()

    Return

ConcatDataToCipherText(ConcatDataToCipherText(encTransform.TransformFinalBlock(plainText,  
0, plainText.Length), salt), algo.IV)

End Function

Public Function Decrypt(cipherText As Byte(), key As Byte()) As Byte()

    Dim cipherTextWithSalt As Byte() = New Byte(0) {}

    Dim encSalt As Byte() = New Byte(0) {}

    Dim origCipherText As Byte() = New Byte(0) {}

    Dim encIv As Byte() = New Byte(0) {}

    SliceCipherTextIntoParts(cipherText, 16, cipherTextWithSalt, encIv)

    SliceCipherTextIntoParts(cipherTextWithSalt, saltSize, origCipherText, encSalt)

    LoadSecureParameters(key, encIv, encSalt)

    Dim decTransform As System.Security.Cryptography.ICryptoTransform =  
algo.CreateDecryptor()

    Dim plainText As Byte() = decTransform.TransformFinalBlock(origCipherText, 0,  
origCipherText.Length)

```
Return plainText
End Function
```

```
Private Function ConcatDataToCipherText(cipherText As Byte(), iv As Byte()) As Byte()
    Dim origLength As Integer = cipherText.Length
    Array.Resize(cipherText, cipherText.Length + iv.Length)
    Buffer.BlockCopy(iv, 0, cipherText, origLength, iv.Length)
    Return cipherText
End Function
```

```
Private Sub SliceCipherTextIntoParts(cipherText As Byte(), secondPartLen As Integer, ByRef
origCipherText As Byte(), ByRef iv As Byte())
    Array.Resize(iv, secondPartLen)
    Buffer.BlockCopy(cipherText, CInt(cipherText.Length - secondPartLen), iv, 0,
secondPartLen)
    Array.Resize(origCipherText, CInt(cipherText.Length - secondPartLen))
    Buffer.BlockCopy(cipherText, 0, origCipherText, 0, CInt(cipherText.Length -
secondPartLen))
End Sub
End Class
End Namespace
```

C#

Code /\*

```
* Copyright (C) 2011, Dextrey (0xDEADDEAD)
* Removing this copyright notice is prohibited without permission from author
* Using this code in your own software product, commercial or not is allowed
*/
```

```
using System;
using System.Collections.Generic;
using System.Text;
```

```
namespace DexCrypt
```

```
{
    class PolyAES
    {
```

```
const int saltSize = 32;
```

```
System.Security.Cryptography.SymmetricAlgorithm algo;
System.Security.Cryptography.RNGCryptoServiceProvider rngAlgo;
byte[] salt;
public PolyAES()
```

```
{
    this.algo = new System.Security.Cryptography.RijndaelManaged();
    this.algo.Mode = System.Security.Cryptography.CipherMode.CBC;

    this.rngAlgo = new System.Security.Cryptography.RNGCryptoServiceProvider();
}
```

```
private void InitializeSecureParameters(byte[] key)
{
    // init rijndael IV
```

```

        this.algo.GenerateIV();
        salt = new byte[saltSize];
        rngAlgo.GetBytes(salt);
        System.Security.Cryptography.Rfc2898DeriveBytes pwDeriveAlg = new
System.Security.Cryptography.Rfc2898DeriveBytes(key, salt, 2000);
        this.algo.Key = pwDeriveAlg.GetBytes(32);
    }

    private void LoadSecureParameters(byte[] key, byte[] encIv, byte[] encSalt)
    {
        this.algo.IV = encIv;
        this.salt = encSalt;
        System.Security.Cryptography.Rfc2898DeriveBytes pwDeriveAlg = new
System.Security.Cryptography.Rfc2898DeriveBytes(key, salt, 2000);
        this.algo.Key = pwDeriveAlg.GetBytes(32);
    }

    public string Encrypt(string plainText, string key)
    {
        return Convert.ToBase64String(this.Encrypt(UnicodeEncoding.UTF8.GetBytes(plainText),
UnicodeEncoding.UTF8.GetBytes(key)));
    }

    public string Decrypt(string cipherText, string key)
    {
        return UnicodeEncoding.UTF8.GetString(this.Decrypt(Convert.FromBase64String(cipherText),
UnicodeEncoding.UTF8.GetBytes(key)));
    }

    public byte[] Encrypt(byte[] plainText, byte[] key)
    {
        InitializeSecureParameters(key);
        System.Security.Cryptography.ICryptoTransform encTransform = algo.CreateEncryptor();
        return
ConcatDataToCipherText(ConcatDataToCipherText(encTransform.TransformFinalBlock(plainText,
0, plainText.Length), salt), algo.IV);
    }

    public byte[] Decrypt(byte[] cipherText, byte[] key)
    {
        byte[] cipherTextWithSalt = new byte[1];
        byte[] encSalt = new byte[1];
        byte[] origCipherText = new byte[1];
        byte[] encIv = new byte[1];

        SliceCipherTextIntoParts(cipherText, 16, ref cipherTextWithSalt, ref encIv);
        SliceCipherTextIntoParts(cipherTextWithSalt, saltSize, ref origCipherText, ref encSalt);
        LoadSecureParameters(key, encIv, encSalt);
        System.Security.Cryptography.ICryptoTransform decTransform = algo.CreateDecryptor();
        byte[] plainText = decTransform.TransformFinalBlock(origCipherText, 0, origCipherText.Length);

        return plainText;
    }

```

```

private byte[] ConcatDataToCipherText(byte[] cipherText, byte[] iv)
{
    int origLength = cipherText.Length;
    Array.Resize(ref cipherText, cipherText.Length + iv.Length);
    Buffer.BlockCopy(iv, 0, cipherText, origLength, iv.Length);
    return cipherText;
}
private void SliceCipherTextIntoParts(byte[] cipherText, int secondPartLen, ref byte[]
origCipherText, ref byte[] iv)
{
    Array.Resize(ref iv, secondPartLen);
    Buffer.BlockCopy(cipherText, (int)(cipherText.Length - secondPartLen), iv, 0, secondPartLen);
    Array.Resize(ref origCipherText, (int)(cipherText.Length - secondPartLen));
    Buffer.BlockCopy(cipherText, 0, origCipherText, 0, (int)(cipherText.Length - secondPartLen));
}
}
}

```

Example (note the BAD key):

```

Code Dim enc As New PolyAES()
Dim dec As New PolyAES()

```

```

Console.WriteLine(enc.Encrypt("Hello, World!", "key123456"))
Console.WriteLine(dec.Decrypt(enc.Encrypt("Hello, World!", "key123456"), "key123456"))
Code PolyAES enc = new PolyAES();
PolyAES dec = new PolyAES();

Console.WriteLine(enc.Encrypt("Hello, World!", "key123456"));
Console.WriteLine(dec.Decrypt(enc.Encrypt("Hello, World!", "key123456"), "key123456"));

```

**Kill?Muvz** - July 16, 2011, 10:32 am

I fucking love you. Seriously.

**W33DY** - July 16, 2011, 10:51 am

Nice,thank you for the Code  
Stars added :)

**hgdfhsdgdgdfdh** - July 17, 2011, 3:17 am

I'm glad for positive comments.

**.CRz** - July 17, 2011, 4:00 am

? yeah, i love your work. and i like the way you code... even if you ignore my pm's ?

**hgdfhsdgdgdfdh** - July 17, 2011, 4:19 am

Yeah only one, small function is better for crypters :P

I prefer writing my code in classes (but I still have some structural problems, things could be written better) because its cleaner.

But no offense, very big thanks for posting the functions.

**Stenography** - July 17, 2011, 4:46 am

oh god ...

Thank you so much !

**Eternal Five** - July 17, 2011, 5:19 am

This is very nice, good job.

**hgdfhsdgdgdfgh** - July 17, 2011, 12:40 pm

Its quite easy to modify it to use any symmetric cipher which is class inheriting from SymmetricAlgorithm class. I'm gonna update it soon to make it easy to select between ciphers.

**chris0147** - July 17, 2011, 2:58 pm

wow nice one. can i encrypt and decrypt the strings?

if so how i can protected them and avoid the encrypted to be cracked?

**hgdfhsdgdgdfgh** - July 18, 2011, 4:14 am

There is direct overload of encrypt function to encrypt strings.

You can encrypt string by:

```
PolyAES polyAes = new PolyAES();  
string encryptedString = polyAes.Encrypt("string to encrypt", "secure encryption key")  
polyAes.Decrypt(encryptedString, "secure encryption key")
```

Have a enough secure key (very long password, or random generated +20 character string) and nobody can crack the string.

**d3c0mpil3r** - July 19, 2011, 6:01 am

If I got this right, when encrypting, you are storing the IV in the encrypted byte array. Later extracting that while decrypting?

Similar to [this](#)?

btw, I really like your work :)

**hgdfhsdgdgdfgh** - July 19, 2011, 6:21 am

The IV is stored in plain format right after the ciphertext ('merged' to it: ciphertext + iv).

And its extracted and then ciphertext is decrypted using it and the key from user.

**chris0147** - July 19, 2011, 1:22 pm

(Jul 18, 2011 04:14 AM)0xDEADDEAD Wrote: [\[CITE\]](#) There is direct overload of encrypt function to encrypt strings.

You can encrypt string by:

```
PolyAES polyAes = new PolyAES();  
string encryptedString = polyAes.Encrypt("string to encrypt", "secure encryption key")  
polyAes.Decrypt(encryptedString, "secure encryption key")
```

Have a enough secure key (very long password, or random generated +20 character string) and nobody can crack the string.

thanks. what about if i can decrypt the strings?

**hgdfhsdgdgdfgh** - July 20, 2011, 7:48 am

With Decrypt function. If your key is good enough nobody can crack the strings without knowing the key.

**chris0147** - July 21, 2011, 4:09 pm

Yes i know that, but i want to know how to decrypt the strings using with aes class?

hgdfhsdgdgdfh - July 22, 2011, 2:11 am

I dont get what you're trying to say. I already said in my earlier post that you can decrypt string with Decrypt function.

Example:

```
Code Dim enc As New PolyAES()  
Dim dec As New PolyAES()
```

```
Console.WriteLine(enc.Encrypt("Hello, World!", "key123456"))  
Console.WriteLine(dec.Decrypt(enc.Encrypt("Hello, World!", "key123456"), "key123456"))
```

X750434 - October 18, 2011, 10:39 am

Grate Work Mate..

Keep Up the Grate Work..

Priv9 Asper Mods - October 18, 2011, 11:01 am

Stop with the incorrect use of the word polymorphic -.-

noober1001 - October 19, 2011, 8:04 am

OMG, this is fucking great.

Thank you so much for this.

chris0147 - December 26, 2011, 3:29 pm

(Jul 18, 2011 04:14 AM)0xDEADDEAD Wrote: [\[CITE\]](#) There is direct overload of encrypt function to encrypt strings.

You can encrypt string by:

```
PolyAES polyAes = new PolyAES();  
string encryptedString = polyAes.Encrypt("string to encrypt", "secure encryption key")  
polyAes.Decrypt(encryptedString, "secure encryption key")
```

Have a enough secure key (very long password, or random generated +20 character string) and nobody can crack the string.

Can you give me a proof that nobody would crack them if i store them in my php page?

hgdfhsdgdgdfh - December 27, 2011, 1:45 am

I cannot proof anything, but I am 99.9999 (add 100 new 9's) % sure that nobody will crack them if you store them on PHP page and your ENCRYPTION KEY is SECURE and you DON'T store the encryption key on PHP page.

"The National Security Agency (NSA) reviewed all the AES finalists, including Rijndael, and stated that all of them were secure enough for U.S. Government non-classified data. In June 2003, the U.S. Government announced that AES may be used to protect classified information"

[http://en.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard#Security](http://en.wikipedia.org/wiki/Advanced_Encryption_Standard#Security)

chris0147 - December 27, 2011, 3:43 pm

Okay, so what do you mean by "your ENCRYPTION KEY is SECURE and you DON'T store the encryption key on PHP page."

is it not safe if i store the encrypted strings in my php page as the cracker can still be able to find them and crack it?

Sorry if i am sound like a noob, but im new into this. :D

**Polymorphic Malewear** - December 29, 2011, 8:16 pm

You do realize this is dynamic not polymorphic, to be polymorphic the code has to physically change.

**hgdfhsdgdgdfdh** - December 30, 2011, 6:04 am

Yes I realise that. Thats why polymorphic was in quotes on title. But now I changed it to prevent confusion...

**darkconcept** - December 30, 2011, 7:48 am

nice man, i was going to whip up some code like this a month or so ago, but forgot about it. you sparked my interest again. like your c# code!

**mundi\_86** - January 3, 2012, 8:43 am

fuck this snip is EPIC!  
Big thx ?

**mundi\_86** - January 7, 2012, 1:41 am

(Jan 3, 2012 09:43 AM)mundi\_86 Wrote: [\[CITE\]](#) fuck this snip is EPIC!  
Big thx ?

here a small demo about the different.

String "Hello Wolrd!" crypt 3x with same password! (PolyAES)  
String "Hello Wolrd!" crypt 3x with same password! (TrippleDES)

[img]<http://img822.imageshack.us/img822/6553/unbenannt1dy.jpg>[/img]

i love this code!!! ?

**JeromeMarshall** - January 7, 2012, 2:11 pm

This looks like it will be very helpful for a lot of people. I have some code much similar, but yours looks a bit cleaner :P

**DragonHunter** - January 7, 2012, 3:20 pm

(Dec 27, 2011 02:45 AM)0xDEADDEAD Wrote: [\[CITE\]](#) I cannot proof anything, but I am 99.9999 (add 100 new 9's) % sure that nobody will crack them if you store them on PHP page and your ENCRYPTION KEY is SECURE and you DON'T store the encryption key on PHP page.

"The National Security Agency (NSA) reviewed all the AES finalists, including Rijndael, and stated that all of them were secure enough for U.S. Government non-classified data. In June 2003, the U.S. Government announced that AES may be used to protect classified information"

[http://en.wikipedia.org/wiki/Advanced\\_Encryption\\_Standard#Security](http://en.wikipedia.org/wiki/Advanced_Encryption_Standard#Security) Well I'm curious what they have TOP SECRET at 2048bit encryption, Area 51 maybe ? ?

**Xer0Hax** - January 8, 2012, 9:35 am

Why no one uses OTP?

Read about it here: [http://en.wikipedia.org/wiki/One-time\\_pad](http://en.wikipedia.org/wiki/One-time_pad)

**hgdfhsdgdgdfdh** - January 8, 2012, 10:41 am

Because of: [http://en.wikipedia.org/wiki/One-time\\_pad#Problems](http://en.wikipedia.org/wiki/One-time_pad#Problems)

**St0rmX** - January 8, 2012, 11:02 am

I Will be using this :)

thanks alot.

**Xer0Hax** - January 8, 2012, 11:29 am

(Jan 8, 2012 11:41 AM)0xDEADDEAD Wrote: [\[CITE\]](#) Because of:

[http://en.wikipedia.org/wiki/One-time\\_pad#Problems](http://en.wikipedia.org/wiki/One-time_pad#Problems)

Implement it properly and it will be fine. Not as simple as copy+pasting a RC4 function, though.

**hgdfhsdgdgdfh** - January 8, 2012, 12:28 pm

Its still more impractical than most block ciphers. You need to transport the key somehow (same size as data).