當麻許的超技八

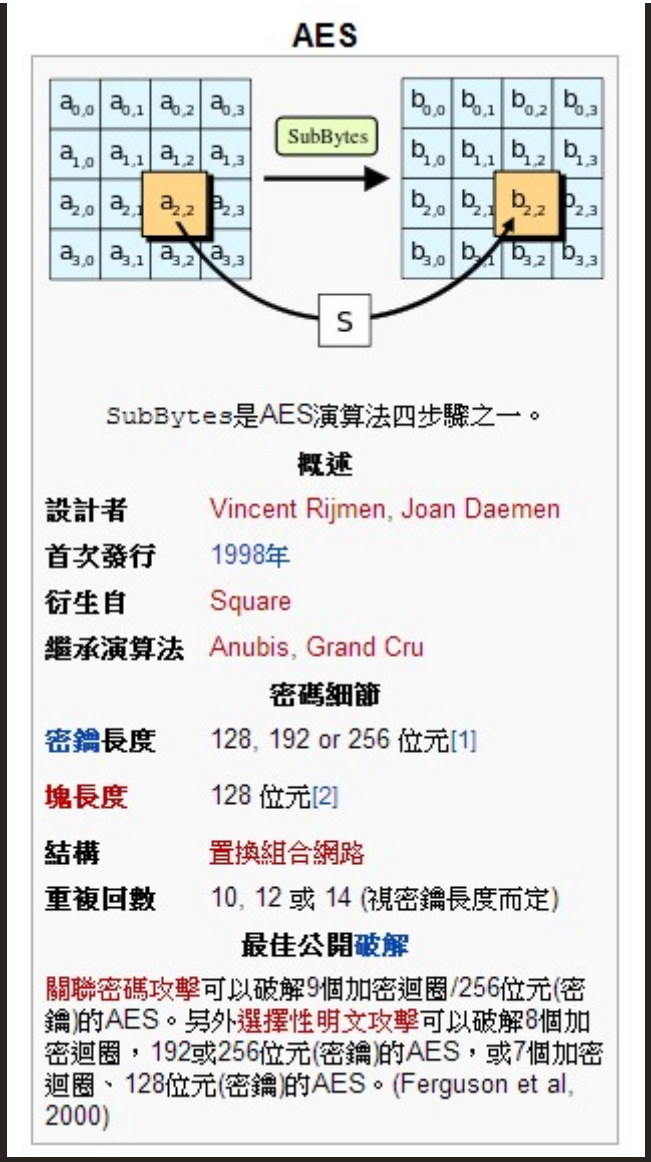
超自然、技術、八卦 - 兼任的偽善者，職業的詐欺師，part-time的故事家，業餘的code generator，不服輸的夢想家，長不大的幼稚鬼，Facebook超自然現象研究社社長，堅持對我來說 就是以剛克剛。

[C#] 與Android共舞-AES 加解密(C# 端)

2013年3月13日

因為最近在弄Android 對於安全行問題會需要用到加密的方法..
看了一下AES 看一下 Wiki 上面說的

> *進階加密標準（Advanced Encryption Standard，AES），在密碼學中又稱Rijndael加密法，是美國聯邦政府採用的一種區塊加密標準。這個標準用來替代原先的DES，已經被多方分析且廣為全世界所使用。經過五年的甄選流程，進階加密標準由美國國家標準與技術研究院（NIST）於2001年11月26日發佈於FIPS PUB 197，並在2002年5月26日成為有效的標準。2006年，進階加密標準已然成為對稱密鑰加密中最流行的演算法之一。*
> *該演算法為比利時密碼學家Joan Daemen和Vincent Rijmen所設計，結合兩位作者的名字，以Rijndael為名投稿進階加密標準的甄選流程。（Rijndael的發音近於 "Rhine doll"）*



其中，有一段比較需要注意的，嚴格地說，AES和Rijndael加密法並不完全一樣（雖然在實際應用中二者可以互換），因為Rijndael加密法可以支援更大範圍的區塊和密鑰長度：AES的區塊長度固定為128 位元，密鑰長度則可以是128，192或256位元；而Rijndael使用的密鑰和區塊長度可以是32位元的整數倍，以128位元為下限，256位元為上限。加密過程中使用的密鑰是由Rijndael密鑰生成方案產生。

所以在設定Key 跟 IV 時都需要符合規範

C# Code - 加密 :

```
/// <summary>
/// 使用AES 256 加密
/// </summary>
/// <param name="source">本文</param>
/// <param name="key">因為是256 所以你密碼必須為32英文字=32*8=256</param>
/// <param name="iv">IV為128 所以為 16 * 8= 128</param>
/// <returns></returns>

public static string EncryptAES256(string source, string key, string iv)
{
    byte[] sourceBytes = Encoding.UTF8.GetBytes(source);
    var aes = new RijndaelManaged();
    aes.Key = Encoding.UTF8.GetBytes(key);
    aes.IV = Encoding.UTF8.GetBytes(iv);
    aes.Mode = CipherMode.CBC;
    aes.Padding = PaddingMode.PKCS7;

    ICryptoTransform transform = aes.CreateEncryptor();

    return Convert.ToBase64String(transform.TransformFinalBlock(sourceBytes, 0, sourceBytes.Lengt

}
```

其中最後，我會使用Base64 輸出變成字串好傳遞.

C# Code - 解密 :

```
/// <summary>
/// 使用AES 256 解密
/// </summary>
/// <param name="encryptData">Base64的加密後的字串</param>
/// <param name="key">因為是256 所以你密碼必須為32英文字=32*8=256</param>
/// <param name="iv">IV為128 所以為 16 * 8= 128</param>
/// <returns></returns>
public static string DecryptAES256(string encryptData, string key, string iv)
{
```

```
    var encryptBytes = Convert.FromBase64String(encryptData);
    var aes = new RijndaelManaged();
    aes.Key = Encoding.UTF8.GetBytes(key);
    aes.IV = Encoding.UTF8.GetBytes(iv);
    aes.Mode = CipherMode.CBC;
    aes.Padding = PaddingMode.PKCS7;
    ICryptoTransform transform = aes.CreateDecryptor();

    return Encoding.UTF8.GetString(transform.TransformFinalBlock(encryptBytes, 0, encryptBytes.Le

}
```

因為使用方便，全部傳遞都使用字串，對外就不看到byte[] 的處理了..

使用方法

C# code :

```
// 32 個英文或數字
string key = "1234567890" +
             "1234567890" +
             "1234567890" +
             "12";

// 16 個英文或數字
string iv = "1234567890" +
            "abcdef";

var encryptResult = EncryptAES256("許當麻", key, iv);
Response.Write("加密後:" + encryptResult + "<br />");
var descryptResult = DecryptAES256(encryptResult, key, iv);
Response.Write("解密後:" + descryptResult + "<br />");
```

加解密結果:

加完密結果為 wifhXJU46ETengY/nA+sGQ==

等等我們到Android 來解解看..當然 Key and IV 都是必須的.

標籤： Android , C#

--

請你暫時把你的勇氣給我　在夢想快消失的時候　讓我的 Code　用力的穿過天空　為愛我的人做一秒英雄　如果這篇文章有幫助到您，簡單留個言，或是幫我按個讚，讓我有寫下去的動力...

0則回應 排序方式: 最舊

新增回應……

Facebook Comments Plugin

較新的文章 首頁 較舊的文章

- Lia (3)
- Linkit7688 (3)
- Linux (4)
- Lucene.net (22)
- MANTIS (2)
- MySQL (2)
- NDK (4)
- Node.js (9)
- NoSQL (8)
- office (6)
- Omnia 7 (2)
- OpenSource (4)
- OrangePi (1)
- PhoneGap (3)
- PHP (2)
- Raspberry Pi (1)
- Regex (5)
- RPi (1)
- RTSP (1)
- SEO (2)
- Silverlight (17)
- SoX (1)
- TFS (1)
- Trick (25)
- UMA (1)
- UWP (5)
- Visual Studio (7)
- Visual Studio Code (1)

AUTHOR

+當麻許
在 Google+ 追蹤當麻