將 iStore 觀念溶入網路空間之設計

金明浩*[a]、張勝欽[ac]、陳建宏[a]、白敏賢[bc]
義守大學資訊工程系[a]
義守大學電子工程系[b]
義守大學電算中心[c]

E-mail
mhjing@isu.edu.tw, chang@isu.edu.tw,d9503001@isu.edu.tw,bair@isu.edu.tw

摘要

高級加密標準(Advanced Encryption Standard，AES)在 2000 年 10 月發布。它是美國國家標準與技術研究院（NIST）向全世界的專家學者公開徵求的加密演算法。這個加密標準已經被證明在大型系統或小型系統上針對軟體或硬體都很容易實做，本論文主要是在 AES 的加密程序過程進行高度多樣化的設計，這樣的作法可以克服一旦加密金鑰遺失時的系統立即不安全性。這種多樣化 AES 加密可以應用在防止網際網路上密碼被竊取攻擊。除此之外，在嵌入式系統上實做 AES 加密也可以溶入 RS 糾錯碼的應用提高系統可靠度。

關鍵字: 加密, 糾錯碼

ABSTRACT

The Advanced Encryption Standard (AES) was decided at Oct, 2000. NIST has called for proposal for this encryption algorithm from the experts in worldwide. This standard turns out that it becomes very flexible for hardware or software implementation on small and large systems. Our research is focus on the generation of higher diversity on the AES system configuration and overcome the immediate dangerous when key is lost. This diversified AES code may apply on the web to prevent the attack from the stealer on the web. Also, an embedded system is build to protect the AES code by using a fast RS error correction code decoder to enhance the system reliability.

Keywords: encryption, error correction code

I. 前言

資料保存一直以來都是被重視的課題，我們常把資料保存於硬碟、光碟、隨身碟等等硬體。但若是硬體被取得，例如設備遺失或送修，裡面的資料就一覽無遺。即便資料使用系統軟體設定密碼才能開啟，如 Microsoft word 軟體密碼保護，但只要透過低階的硬碟存取程式，仍可讀取裡面資料，因為資料本身沒有加密。將電腦設備放在很安全的環境仍有資料保存風險，因為硬體會損壞，電腦硬體本身都有使用年限，所以資料就可能因著硬體的損壞而遺失。

高度的資料加密可以讓資料得到私密性，資料加入容錯編碼則可以提高資料保存的可靠度，本論文提出一個結合這兩種資料演算的資料保存方法，將電腦數位資料在保存時先行加密演算，然後再加入容錯編碼。而容錯的部份不僅是單純的編入資料容錯碼，我們還提出一個資料分散式的保存方式，將編碼後的資料與以切割並分散於各種不同的網路空間，更提高資料保存的可靠度。並且為了方便使用者可以隨時使用，我們將該系統安裝於 USB 隨身碟，只要使用隨身碟內的該系統，就可以很容易將檔案完成加密及容錯分散儲存於網路上，要用的時候再從網路上取回，使用者不需帶著大量資料到處跑，因為資料都是經過加密，使用者也不需擔心資料遭竊取。

II. 系統分析

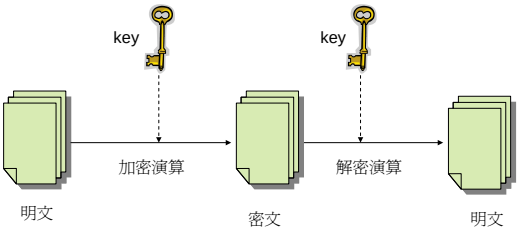
對私密資料的保護最常見的就是加密，例如網路通訊上常用的 HTTPS、SSL 或 SSH 等。目前學術或商業界有許多加密的演算法可以使用，但有些是不公開的，或是要付費才能使用。即便是好的加密系統，一旦金鑰遺失，私密資料就容易遭破解。資料加密後也必須妥善保存，因為資料加密之後的狀態是很脆弱的，只需 1 個 bit 的錯誤，就能使資料無法完全回復讀取，因此對資料的管理而言，資料容錯也是必須要功能。但是資料容錯之後，如果放置於單一地點或設備，難免有天災人禍或是設備老舊損毀的風險，所以資料若能分開存放於不同地點，就可以增加資料保存可靠度。

現今網際網路（Internet）盛行，我們希望能夠隨時透過網路存取自己私密性資料，而這些資料都是加密過並且容錯分散於不同地點，這會對現代資訊使用者是一個極方便的好處。如此的設計理念是以保護使用者不共享的私密資料及隨時使用為目的，以下分別針對本系統所採用的加密與容錯演算分析如下：

A. 加密系統

密碼學（Cryptography）在網際網路的應用相當廣泛，資料在保存或傳輸過程中的加密需求愈來愈普遍。IBM 公司於 1977 年發表「資料加密標準」DES(Data Encryption Standard)的加密系統，他的速度快，安全性也高，到目前仍在使用中。但是就電腦軟硬體的演變與進步，該加密系統的歷史已經算很久了，有許多針對 DES 破解的研究不斷被提出，縱使後來有改良型的 3DES，但是基本的演算法架構仍未改變，就現今的科技環境而言，安全性已經大不如前。有鑑於此，美國國家標準與技術研究院（NIST）於 1997 年公開徵求進階加密標準（Advanced Encryption Standard,AES），最後選出 Rijdael 加密演算法為 AES，該演算法由比利時密碼學家 Joan Daemen 和 Vincent Rijmen 所設計，並以兩人的名字命名。AES 目前已經成為密碼學相當流行的演算法。

密碼學的應用的記載在距今兩千多年前就開始，雖然經過不斷的演變，但傳統上基本的主要作法不外乎「取代」、「換位」、「混淆」、「散置」。「取代」就是把原資料的字元用另一字元表示，如字母 A 就由 M 取代，B 由 N 取代。「換位」的作法是將原資料順序排列變換，如原資料為 BOOK，換位後為 KOOB。「混淆」就是在原來的資料加入其他的資料使其更混亂，原資料為 BOOK，混淆後成為 BXOYOZK。「散置」則是將資料加密後放置於不同的位置達到變亂效果。如使用特定的演算法規則再結合「取代」、「換位」、「混淆」、「散置」的計算，就可以成為一個加密演算法，而這些特定演算法都需要有金鑰的機制，就好像一般人在日常生活習慣中，會把重要資料或貴重的財物、金錢等放在保險箱鎖住，需要鑰匙或密碼才能開啟。加解密的基本作法如下圖所示：



不同於 DES 加密演算法承襲自 Feistel 架構，AES 的加密演算法是建立在有限體（Finite Field）數學 GF(2^8)架構上，GF 是指 Galois Field，他是 19 世紀法國數學家 Galois 所提出一種代數運算結構體，一般都是以多項式（polynomial）來表示，並且必須至少存在一個不可分解多項式（irreducible polynomial）來作為該有限體運算的模（modulo）。AES 加密的單位區塊大小為 128bits，將 128bit 以矩陣方式依序排列做加密步驟演算。加密金鑰 key 的長度有三種，分別為 128bits、192bits、256bits。AES 加密主要的 4 個步驟為 AddRoundKey、SubBytes、ShiftRows、MixColumns。以 128bits 的 key 為例，其加密第 1 次做完 AddRoundKey 後，再分別以 9 次回合運算做 SubBytes、ShiftRows、MixColumns、AddRoundKey。結束後再依序做 1 次 SubBytes、ShiftRows、AddRoundKey。

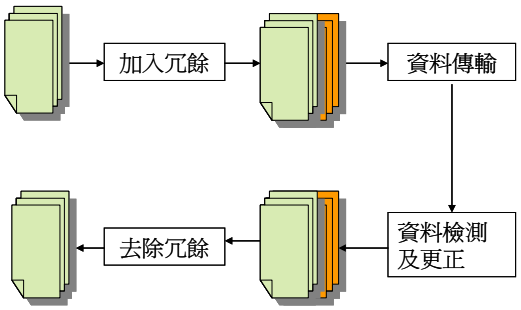
AddRoundKey 就是將擴充後的 key 值與資料的 128bits 矩陣加密單位做 XOR 運算的結果為新值，例如假設某資料矩陣值 11011011，鍵值矩陣為 10110010，則 AddRoundKey 後的結果為 01101001。SubBytes 即是上述的「取代」運算，用一個表的值取代原值，一般都稱這表為 S-box，該表的產生是由有限體代數 GF(2^8)的乘法反元素（multiplication inverse）計算出每個取代的值，其中使用的不可分解多項式為 f(x)= $x^8 + x^4 + x^3 + x + 1$，如果不考慮記憶體空間，也可以先把這 256 個值作成一個查表，每個資料值就由著個表的值取代。S-Box 是該加密演算法安全性的一個重要關鍵，他是一個非線性的組合，必須要能抵擋各樣的密碼分析攻擊。ShiftRows 則是做資料位移運算，128bits 的加密單位資料矩陣依序排成 4x4，第一列不位移，第二列向左位移 1 個單位，第三列向左位移 2 個單位，第四列向左位移 3 個單位。MixColumns 是把 4X4 矩陣資料的每行個別值依序作為 1,

x，x^2, x^3的係數，形成一個 GF(2^8)的多項式，此多項式與固定多項式 $\{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$ 在模（modulo）多項式 x^4+1下做有限體乘法運算，結果的值成為該矩陣行的值。

B. 容錯系統

數位電子訊號傳輸處理過程中很容易因外界環境緣故被干擾，因此很多錯誤檢測機制用來確保資料傳輸正確無誤，其編碼方法就是在原資料加入資料冗餘。其編解碼都是經由相對應的容錯編碼方法計算而得，如同位元檢查方法，假設資料為 10010110，假設作奇位元校驗，由於該資料每個位元資料之 XOR 計算結果為 0，所以要加入位元 1 為資料冗餘，結果為 100101101，每個位元資料之 XOR 計算結果為 1，達到奇位元校驗，一旦資料傳輸後發生錯誤，例如變為 100101111，其每個位元 XOR 計算結果為 0，便可得知資料有誤，必需更正或重新傳輸。上述方法相當簡單，但也有比較複雜的應用，例如 CRC(Cyclic Redundancy Check)檢測機制，其方法是將數位資料以多項式表示，例如 11010011 表示為 $x^7+x^6+x^4+x+1$，再以選定的多項式（例如 x^4+x+1）作為除數去除以原傳輸資料，以餘數為資料冗餘，將原資料增加餘數位元後當作傳輸資料，資料經傳輸後，接收端以同樣選定的多項式為除數計算，若可以整除，則表示資料無誤，若不能整除則表示資料傳輸後有錯誤。在網路傳輸運用例如以太網路(Ethernet)的傳輸是使用 CRC32 作為封包資料檢測，若資料錯誤則必須重新傳遞封包以確保資料傳輸無誤。

上述的資料傳遞僅有檢查錯誤，如果有錯誤則須重新傳遞資料，假使我們要求能夠某種程度的更正錯誤，就必須使用其他的容錯編碼，資料傳輸使用錯誤更正編碼的示意圖如下：



常用到的錯誤更正編碼有漢明碼（Hamming Code），以漢明碼（7,4）為例，資料長度為 4 bits，編碼後的長度為 7 bits，也就是加了 3 bits 的資料冗餘，此（7,4）編碼可以更正 1 bit 錯誤。例如假設資料為 1011，經漢明碼編碼後的資料為 1011010。

如果要增加容錯碼錯誤更正的數目，相對的就必須要增加編碼的資料冗餘長度，假設編碼長度為 n，資料長度為 k，則 k/n 表示這個編碼的碼率（code rate），在相同的容錯條件下，碼率愈高，就表示這個編碼效率愈好。本論文所提出的容錯編碼方式是採用 Reed Solomon Code（以下簡稱 RS Code），此碼於 1960 年代被發表出來。該編碼的數學演算也是建構在有限體數學 $GF(2^m)$ 的代數運算上，不同於漢明碼以位元 bit 為編碼單位，該編碼是以模組（symbol）或區塊為單位。$GF(2^m)$ 裡面的 m 表示這個編碼的模組區塊長度，例如 m=8 表示這個 RS Code 的編碼模組為 8 bit = 1 byte，這在電腦的運用上就很方便，因為電腦的計算通常都是以 byte 為單位。在相同的碼長輸出入條件之下這樣的編碼方式與位元為單位編碼方式比較起來，可以獲得較大的編碼效率。

III. 系統研究

A. DAES加解密研究：

由於密碼學是用來保護資料的私密性，難免會招來許多的密碼破解或攻擊行為，而且在嘗試破解該資料時都會先研究該密碼的加密演算碼原理，並且如果系統長時間一直使用同樣的加密演算法，長時間下來能因人為疏失而增加被破解機會。通常駭客最想破解的就是密碼，也就是加密金鑰，如果金鑰一旦被獲得，又知道這是某個加密演算法，如此加密的資料就等於被破解。在做加密時一般人有時密碼（金鑰）為了方便記憶不會設定很複雜而容易被破解，因此本論文為了避免加密金鑰遺失所造成私密資料被破解，提出另外的解決的方式，不是針對金鑰的保存，而是去變化 AES 的加密步驟，如此則每次的加密過程或是不同使用者的 AES 加密演算法都有變異性，這樣即便竊取得金鑰，使用標準的 AES 演算去解密就無法解開這個用變異性 AES 加密的資料。

AES 每回合加密有四個主要的參數，分別是：不可分解多項式、SubBytes 矩陣轉換、ShiftRows 位移向量大小、MixColumns 所用的固定多項式。我們運用數學的代數及電腦演算法使這四個參數在加密過程產生變異性，我們稱這樣變化的 AES 加密為 Diversified AES 或簡稱 DAES，以下將分別敘述：

AES 要產生有限體 GF(2^8)所使用的不可分解多項式為 $x^8+x^4+x^3+x+1$，但是在有限體 GF(2)當中可以找到 30 個不可分解多項式，如果使用不同的不可分解多項式，就可以在計算乘法反元素時產生不同的 S-Box 取代值，即使用相同的金鑰加密，產生的密碼也會因不同的不可分解多項式而有不同的結果

在計算 SubBytes 的矩陣轉換計算（Affine transformation）時，所用到的矩陣如下：

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{pmatrix}$$

該矩陣會存在一個反矩陣以便在解密時得到原值，只要能找到任何 8X8 的矩陣具備存在反矩陣，則也可以作為 SubBytes 的矩陣。

ShiftRows 是在狀態矩陣做位移運算，AES 依據矩陣的排列有固定的位移格式，但是我可以不用按照他的位移方式做，用一套自訂的位移規則達到混亂編碼的目的，在解密時按照自訂的返回位移方法解密即可。

在 MixColumns 步驟也可以找到不同的多項式，AES 做有限體乘法是使用固定的多項式 f(x)= $\{03\}x^3+\{01\}x^2+\{01\}x+\{02\}$，底下我們列出幾個可以作為 MixColumns 步驟使用的多項式：

$\{01\}x^3+\{01\}x^2+\{02\}x+\{03\}$

$\{02\}x^3+\{07\}x^2+\{01\}x+\{05\}$

$\{03\}x^3+\{0b\}x^2+\{01\}x+\{08\}$

$\{04\}x^3+\{0e\}x^2+\{07\}x+\{0c\}$

$\{05\}x^3+\{03\}x^2+\{06\}x+\{01\}$

$\{06\}x^3+\{08\}x^2+\{04\}x+\{0b\}$

$\{07\}x^3+\{0d\}x^2+\{05\}x+\{0e\}$

$\{08\}x^3+\{01\}x^2+\{0b\}x+\{03\}$

$\{09\}x^3+\{09\}x^2+\{08\}x+\{09\}$

B. RS Code編解碼研究：

RS Code 特性是可以依照自訂的需求，在設計上加入任何長度的資料冗餘。假設每筆模組資料為 m 位元，其編碼方式為（n ,k），n 表示編碼後模組長度，k 表示原資料模組長度，$n=2^m-1$，n - k = 2t = d - 1，d 表示編碼之間的距離，t 表示可以更正的隨機錯誤（random error），2t 表示可以更正的刪除錯誤（erasure error）。假設 RS Code（n ,k），原資料多項式為 m(x)，編碼資料冗餘為 p(x)，生成多項式（generator polynomial）為 g(x)，我們要把資料冗餘加在原資料後面，所以要位移 x^{n-k}，可以寫成如下的表示式：

x^{n-k}m(x)=q(x)g(x)+p(x)

p(x)= x^{n-k}m(x) modulo g(x)

編碼後的多項式 u(x)= x^{n-k}m(x)+p(x)

舉例 m=3，t=2，每個模組長度為 3 bits，則其該編碼建立在 $GF(2^3)$，其編碼模組長度為 $2^3-1=7$，該 RS Code（7,3）編碼可以解 2 個模組的錯誤，例如原資料 3 個模組為 010,110,111，採用有限體乘法質多項式（primitive polynomial）為 x^3+x+1，

生成多項式

g(x)= $(x-a)(x-a^2)(x-a^3)(x-a^4)$

a,a^2,a^3,a^4 表示他的根(root)，展開經數位化簡後得 g(x)= $x^4+a^3x^3+x^2+ax+a^3$

編碼後 u(x)=

$1+a^2x\ +a^4x^2+a^6x^3+a\ x^4+a^3x^5+a^5x^6$

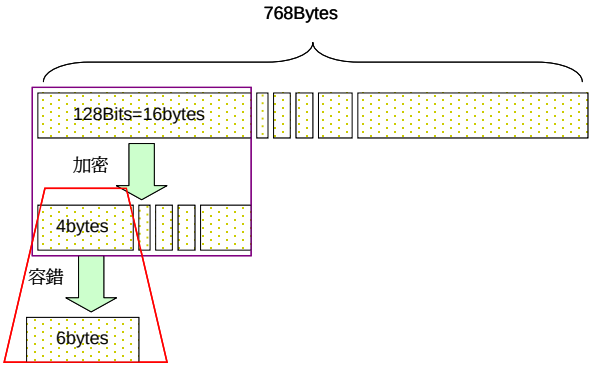
其中的係數是由質多項式 x^3+x+1 乘法表生成，各係數即為編碼後的碼字（codeword），7 個係數如下：100, 001, 011, 101, 010, 110, 111，可更正 2 個模組的隨機錯誤。

本系統運用 RS Code 的模組編碼特性，將資料做容錯分散切割。採用 RS(15,13)編碼方式各縮減 9 碼成為（6,4）碼，模組 m=4，其每一筆資料(symbol)用 4 bits 表示。在解容錯時，因為是已知錯誤位置，所以該碼可以更正 6 – 4 = 2 個刪除錯。

C. 檔案處理核心研究：

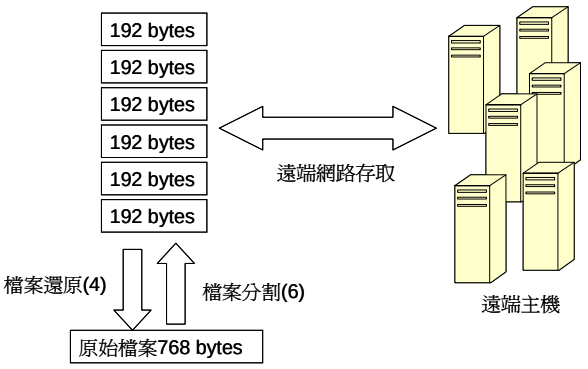
結合上述加密及容錯之演算法，整合發展出一套對硬碟檔案的加密與容錯核心計算元件，分別採用 DAES 與 RS Code。既然要將檔案分散保存，就必須要將檔案切割。一般在做檔案切割時，會先預設切割檔案數目，再按照檔案容量計算依序讀取每個切割檔的位元數，若按此方式處理則檔案要取回時每個切割檔都必須存在，並且切檔案的排列方式也不會太複雜。本系統則是運用 RS Code 模組編碼特性，直接在資料位元組經容錯編碼演算生成冗餘編碼位元組，每個位元組都單獨生成切割檔。以上述提到 RS(6 , 4)編碼方式，將 4 個位元組編入 2 個資料冗餘成為 6 個位元組，可以更正 2 個位元組的錯誤，也就是只要任 4 個位元組就可以還原檔案，不需要全部檔案。以這樣演算法方式，本系統將檔案打散後分成 6 個檔案，其中 2 個是容錯切割檔，只要任 4 個切割檔就可以生成原來的檔案。

本系統的檔案處理部分，先以一次讀取 768 bytes 為單位做處理，再分成 48 個 16bytes(取 AES 加密單位 128 bits = 16 bytes)，先行加密後再進行容錯分割，加密完後的資料長度不變仍為 128 bits，再以每個 4 bytes 為容錯處理單位，每個單位容錯處理後加入資料冗餘 2 bytes 總共為 6 bytes，檔案處理流程如下圖：。



使用者將 USB 隨身碟插入電腦執行程式後，需輸入自行設定的帳號、密碼，該個人的帳號密碼經程式轉換做為隨身碟加密所需的 AES 金鑰以及參數設定。將資料檔案放入隨身碟後，若先前使用者有指定遠端主機之 IP 位址或網域名稱，則檔案經加密及容錯分割後會上傳到遠端不同主機，隨身碟內會保留 1 個檔案分割，若未指定遠端主機之 IP 位址或網域名稱，則檔案仍加密及容錯分割在隨身碟之內，一樣保有加密及容錯預設功能。

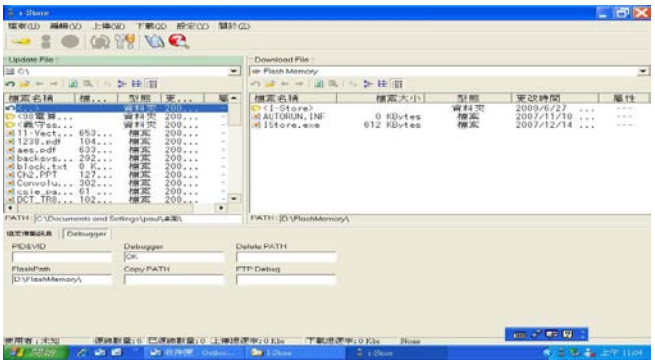
假設檔案容量為 768 bytes，做 6 個切割檔，原先每個切割為 128 bytes，經加密及容錯處理之後變成 6 個 192 bytes 共 1152 bytes，如下圖所示：



IV. 系統實做

本系統是以 Borland C++ builder 程式開發完成，系統設計為 windows GUI 操作介面方便操作使用，包含使用者驗證、檔案管理、網路協定遠端存取、檔案編解碼核心等功能，所有功能都使用 C++語言設計成類別，其中的檔案編解碼處理核心包含加密、容錯及解密、解容錯，都是以標

準 C++語言開發完成，該類別程式庫只要是標準 C++語言編譯器均可正常編譯，可移植到不同作業系統，例如在 Linux 作業系統上使用 GCC 也可以正常編譯該類別程式庫，再以加密及容錯為基礎發展在不同作業系統上不同功能的應用程式。 本系統執行畫面如下圖：



V. 結論

本論文結合密碼學與資料更正碼學原理，實做一個資料分散的網路儲存系統，加密使用 AES 加密演算的技術，並在其加密過程中修改相關參數使之產生與標準 AES 不同的加密結果，此種變異性可以提高系統安全性。使用 RS Code 的模組容錯編碼特性，以高碼率的容錯編碼產生資料分割檔。這些資料分割檔透過網路分散於不同主機或空間，大大提昇資料保存可靠度及使用的便利性。本系統所發展出的資料分散網路儲存技術，此種構想不但使資料的私密性獲得保護，並且讓資料保存的可靠度大大提升。使用者在 USB 隨身碟上安裝此程式，就能夠很方便並安全的的使用自己資料。並且資料存放於網路主機上，自己的 USB 隨身碟不需儲存大量資料，實現龐大資料隨身帶著走、即時存取的構想。

參考文獻

[1] 2007 全國大專校院創意實作競賽http://twcia-contest.isu.edu.tw/

[2] Ming-Haw Jing, Zih-Heng Chen, Jian-Hong Chen, Cheng-Yi Wu, "Design of Simple and High Speed VLSI Core for the Protection of Mass Storages," *IEEE Asia-Pacific Conference on Circuits and Systems*, Macao, China, 2008.12

[3] Ming-Haw Jing, Jian-Hong Chen, Zih-Heng Chen, "Diversified Mixcolumn Transformation of AES," *Sixth International Conferences on Information, Communications and Signal Processing*, Singapore, 2007.12

[4] Ming-Haw Jing, Zih-Heng Chen, Jian-Hong Chen, and Yan-Haw Chen, "Reconfigurable System for High-Speed and Diversified AES Using FPGA," *Microprocessors and Microsystems*, vol. 31, no. 2, pp. 94-102, 2007.03

[5] Ming-Haw Jing, Zih-Heng Chen, and Jian-Hong Chen, "High-Speed Multiplicative Inversion in Finite Field Using Normal Base," *Engineering Science & Technology Bulletin, NSC*, vol. 87, pp. 21-23, 2006.08

[6] 金明浩, 羅坤松, 黃孟逢, 陳信宏, 張國豐, "一個提昇小型容錯系統可靠度的設計," 義守大學學報, vol. 4, pp. 161-173, 1997.08

[7] 金明浩, 陳延華, 張耀祖, 張肇健, "A New VLSI for Implementing the Multiplication and Inverse in the RS-Code," 1999 第四屆多媒體技術及應用學術研討會, pp.304, 高雄

[8] M.H. Jing, Y.H. Chen, J.E. Liao, "A Fast Error and Erasure Correction Algorithm for a Simple RS-RAID," Int. Conf. On Info-tech and Info-net, Beijing, Oct. 29, 2001, pp. 333-8

[9] http://ptgmedia.pearsoncmg.com/images/art_sklar7_reed-solomon/elementLinks/art_sklar7_reed-solomon.pdf

[10] FIPS197 ,http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf