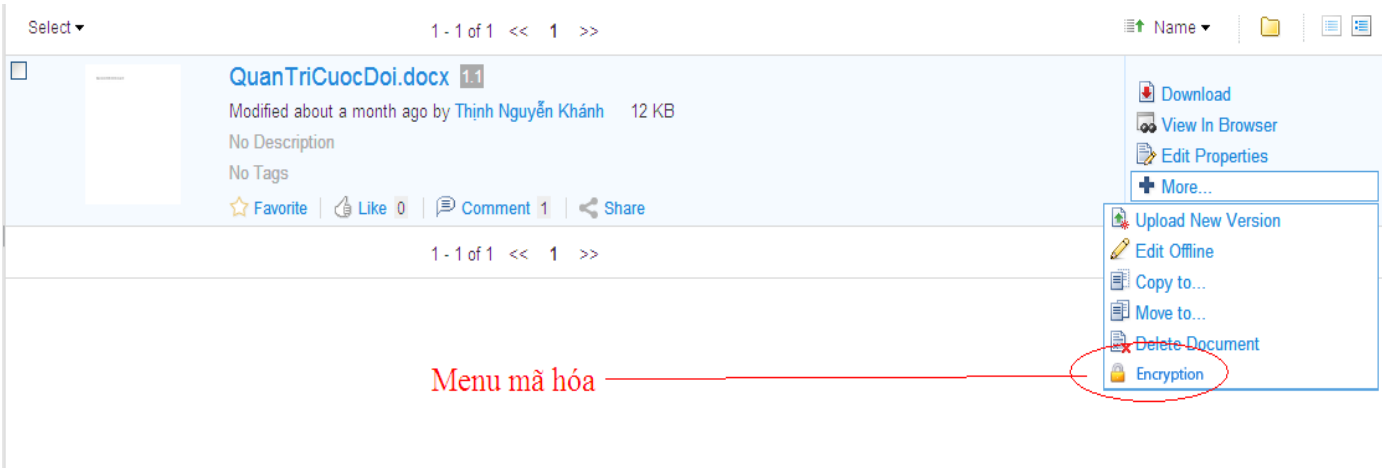
Tài liệu Thiết kế

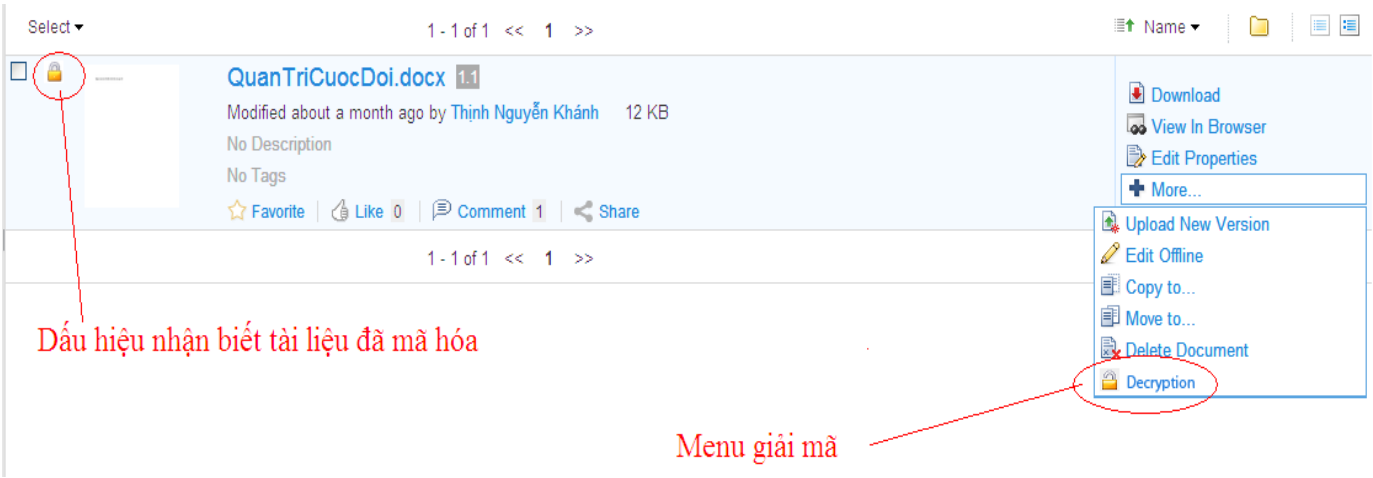
Nội dung:

Thiết kế giao diện người dùng

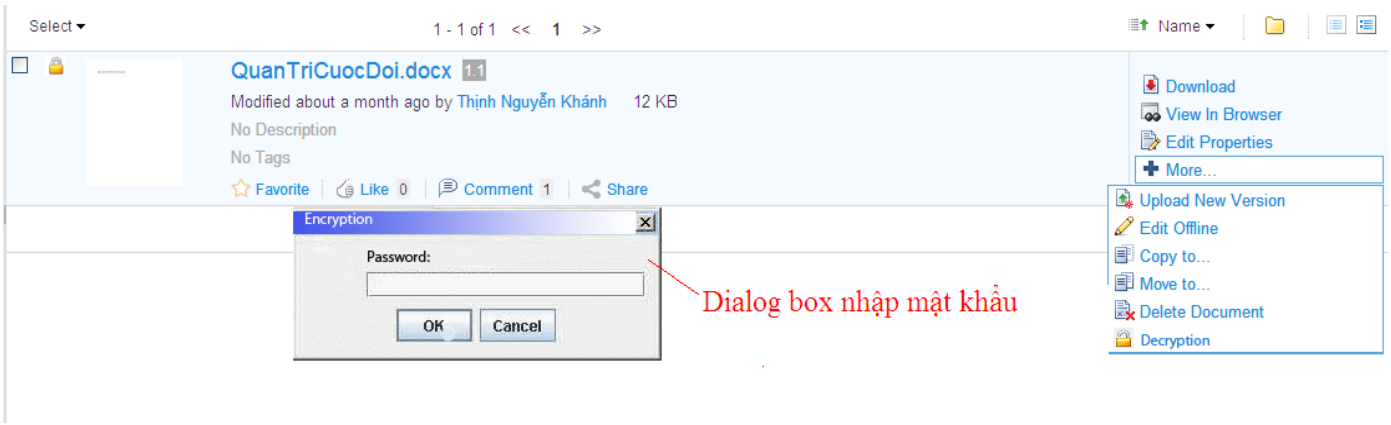
Mã hóa



Giải mã



Form nhập mật khẩu



Thiết kế kiến trúc tổng quát

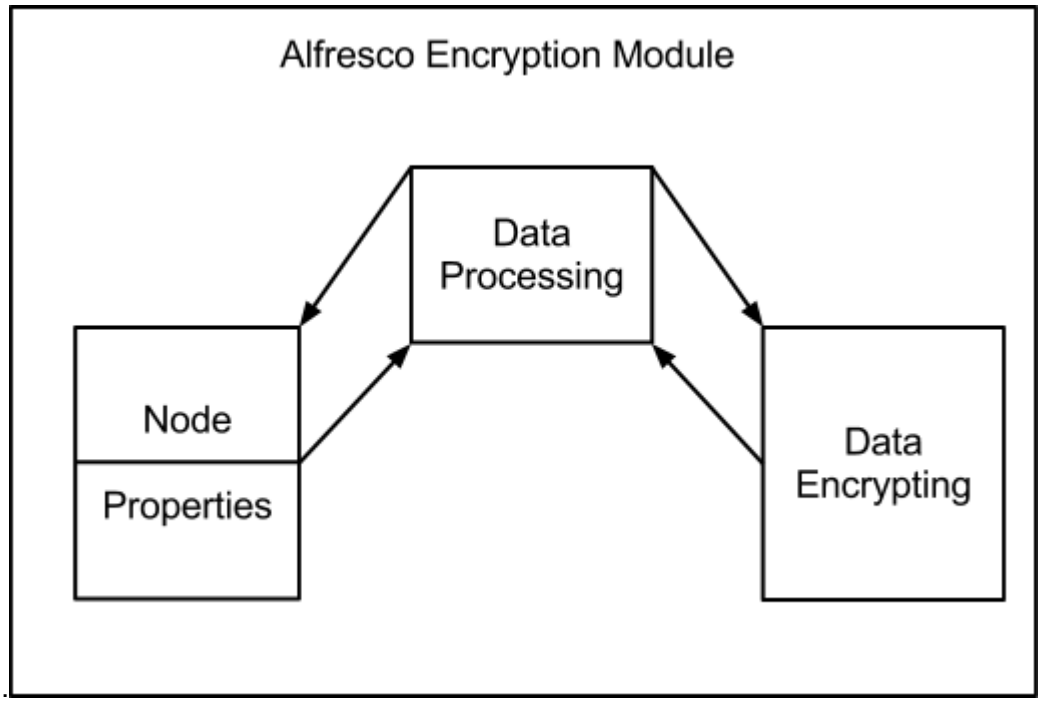
Kiến trúc module mã hóa

Trong Alfresco thì mọi dữ liệu đều ở dạng một “node”. Phần dữ liệu đó được xác định dưới dạng DataType.

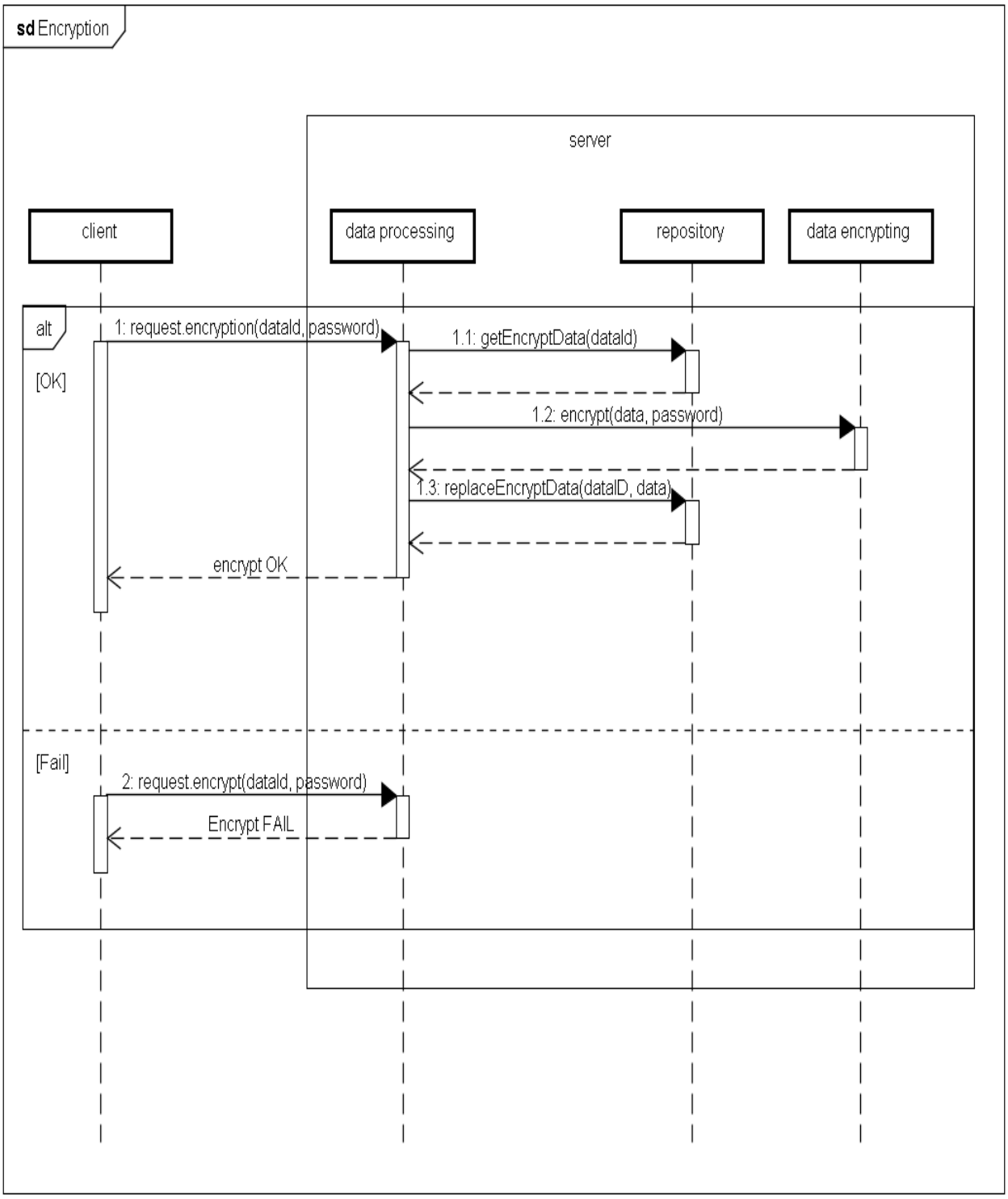
Module mã hóa bao gồm hai phần chính Data Processing và Data Encrypting

- Bộ phận thứ nhất Data Processing: Đọc dữ liệu từ repository và ghi dữ liệu trở lại, thay đổi thuộc tính dữ liệu.
- Bộ phận thứ hai Data Encrypting: Mã hóa và giải mã dữ liệu.

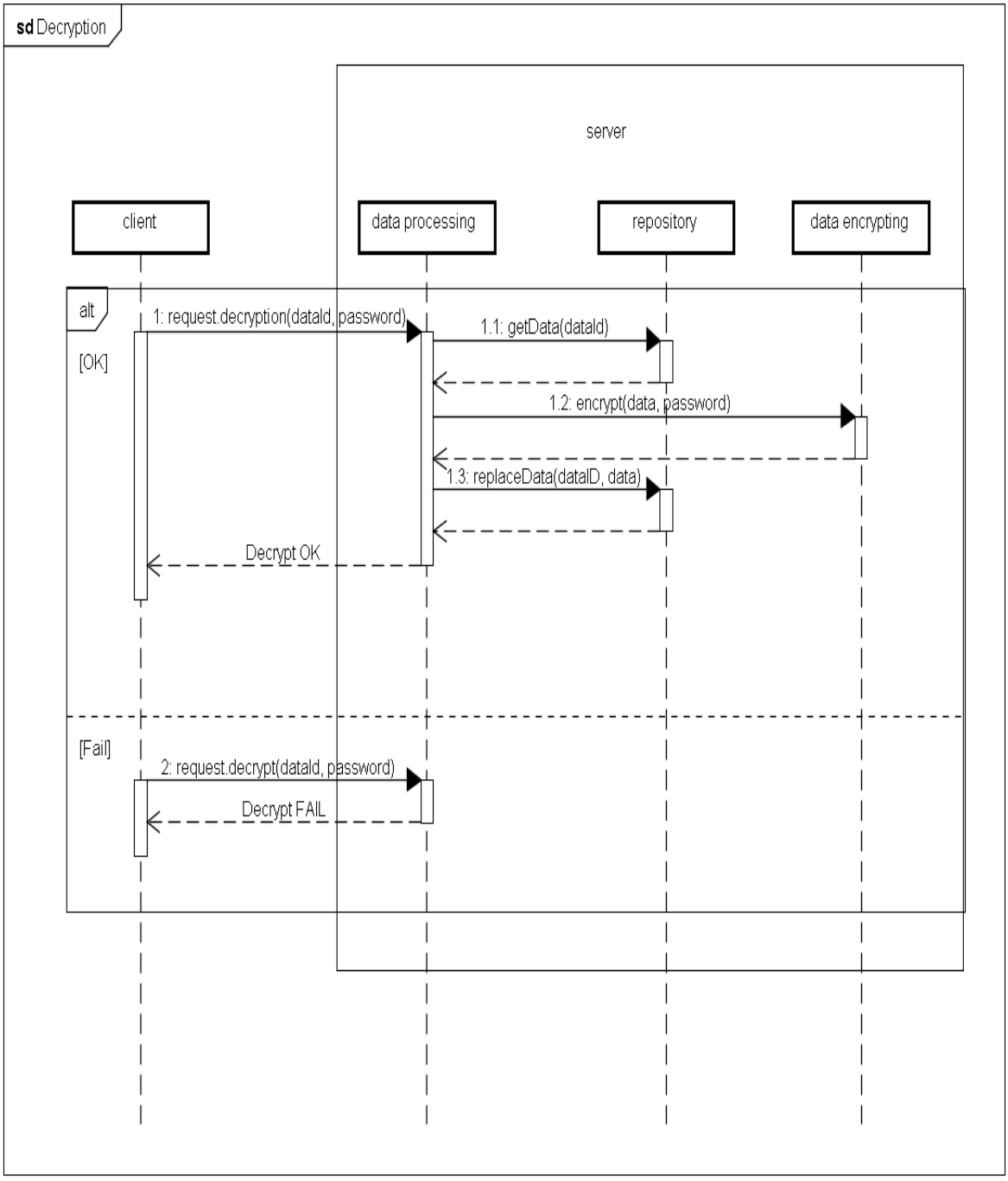
Sơ đồ kiến trúc tổng quan:


Alfresco Encryption Module
Data Processing
Node
Properties
Data Encrypting

Sơ đồ mã hóa


sd Encryption
server
client
data processing
repository
data encrypting
alt
[OK]
1: request.encryption(dataId, password)
1.1: getEncryptData(dataId)
1.2: encrypt(data, password)
1.3: replaceEncryptData(dataID, data)
encrypt OK
[Fail]
2: request.encrypt(dataId, password)
Encrypt FAIL

Sơ đồ giải mã



Thiết kế thuật toán AES

Giới thiệu

AES (Advanced Encryption Standard) là tiêu chuẩn mã hóa theo thuật toán mã hóa đối xứng Rijndael. Tiêu chuẩn được chính phủ Mĩ và NIST (U.S National Institute of Standard and Tecnology) công nhận làm tiêu chuẩn liên bang. Ngày nay tiêu chuẩn mã hóa AES được sử dụng rộng rãi trong nhiều lĩnh vực.
Ngoài ý nghĩa tiêu chuẩn, AES được đề cập tới là thuật toán mã hóa các khối dữ liệu 128 bits bằng các khóa độ dài 128, 192 hoặc 256 bits tương ứng với AES-128, AES-192 và AES-256.
Trong bản thiết kế đặc tả thuật toán bao gồm các nội dung sau:

1. Định nghĩa các khái niệm, kí hiệu và hàm
2. Mô tả thuật toán
3. Các vấn đề khi cài đặt

Định nghĩa, khái niệm và kí hiệu

Thuật ngữ

AES	Advanced Encryption Standard
Affine transformation	Phép biến đổi gồm phép nhân với một ma trận sau đó cộng với một vector
Array	Tập các thực thể được đánh số liệt kê
Bit	Giá trị nhị phân 0 hoặc 1
Block	Chuỗi bits nhị phân gồm input, output, State và Round Key. Độ dài của chuỗi là số bits nó chứa. Block có thể xem như Array của bytes.
Byte	Một nhóm 8 bit xem như một thực thể hoặc như Array của 8 bits đơn
Cipher	Chuỗi các biến đổi bản tường minh (plain text) thành bản mã hóa(cirphertext) dùng Cipher Key
Cipher Key	Khóa mã hóa bí mật được dùng bởi Key Expansion nhằm tạo ra tập các Round Keys:co thể xem như Array của bytes có 4 hàng và Nk cột.
Ciphertext	Dữ liệu đầu ra từ Cipher hoặc đầu vào Inverse Cipher
Inverse Cipher	Chuỗi các biến đổi biến bản mã hóa (ciphertext) thành bản tường mình (plaintext) dùng Cipher Key.
Key Expansion	Các bước dùng để tạo ra chuỗi các Round Keys từ Cipher Key
Plaintext	Dữ liệu đầu vào của Cipher hay đầu ra của Inverse Cipher
Rijndael	Thuật toán mã hóa cơ sở của AES
Round Key	Các giá trị nhận ra từ Cipher Key bằng các sử dụng Key Expansion.

	Được áp dụng với State trong Cipher và Inverse Cipher
State	Mã hóa trung gian cho kết quả là Array của bytes có 4 hàng và Nb cột
S-box	Bảng thế phi tuyến sự dụng trong biến đổi thế byte và trong Key Expansion để thế 1-1 một giá trị byte
Word	Nhóm 32 bits xem như thực thể đơn, hay Array 4 bytes

Hàm, tham số và các kí hiệu

AddRoundKey()	Phép biến đổi trong Cipher và Inverse Cipher ở đó Round Key được thêm vào State dùng phép toán XOR. Độ dài của Round Key bằng kích thước của State.
InvMixColumns()	Phép biến đổi trong Inverse Cipher là ngược của MixColumns
InvShiftRows()	Phép biến đổi trong Inverse Cipher là ngược của ShiftRows()
InvSubBytes()	Phép biến đổi trong Inverse Cipher là ngược của SubBytes()
K	Cipher Key
MixColumns()	Phép biến đổi trong Cipher lấy tất cả các cột của State và trộn dữ liệu của nó một cách độc lập cho ra một cột mới
Nb	Số cột trong State, trong chuẩn này thì Nb=4
Nk	Số cột trong Cipher Key, trong chuẩn này Nk có thể là 4, 6 hoặc 8
Nr	Một số vòng là chức năng của Nk và Nb được cố định. Trong chuẩn này Nr = 10, 12 hoặc 14
Rcon[]	Một Word Array không đổi quay vòng
RotWord()	Hàm dùng Key Expansion nhận 4 byte và hoán vị vòng
ShiftRows()	Biến đổi trong Cipher xử lí State bằng dịch vòng ba cột cuối của State với offsets khác
SubBytes()	Biến đổi Cipher xử lí State bằng phép thế phi tuyến (S-box) lên các bytes của Start độc lập
SubWord()	Hàm dùng trong Key Expansion nhận 4 bytes đầu vào và dùng S-box cho ra một word

Mô tả thuật toán

Trong thuật toán AES, độ dài input block , output block và State là 128 bits. Nó được thể hiện qua Nb = 4 là số 32-bit word trong State.
Trong thuật toán AES, độ dài Cipher Key, K, là 128, 192, hay 256. Độ dài key được thể hiện bởi Nk = 4, 6, hay 8 là số 32-bit words trong Cipher Key.
Trong thuật toán AES, số vòng (rounds) được trình diễn suốt quá trình thực thi thuật toán phụ thuộc vào độ dài key. Số vòng được biết diễn bởi Nr. (Nr= 10 khi Nk= 4, Nr= 12 khi Nk= 6, và Nr= 14 khi Nk= 8.)
Trong cả Cipher và Inverse Cipher, thuật toán AES sử dụng hàm round, được tạo ra từ các phép biến đổi bytes:

1. Byte substitution using a substitution table (S-box)
2. Shifting rows of the State array by different offsets
3. Mixing the data within each column of the State array
4. Adding a Round Key to the State.

Cipher

```
Cipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
      byte state[4,Nb]
      state = in
      AddRoundKey(state, w[0, Nb-1])
      for round = 1 step 1 to Nr-1
            SubBytes(state)
            ShiftRows(state)
            MixColumns(state)
            AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
      end for
      SubBytes(state)
      ShiftRows(state)
      AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1])
      out = state
end
```

Inverse Cipher

```
InvCipher(byte in[4*Nb], byte out[4*Nb], word w[Nb*(Nr+1)])
begin
      byte state[4,Nb]
      state = in
      AddRoundKey(state, w[Nr*Nb, (Nr+1)*Nb-1]) // See Sec. 5.1.4
      for round = Nr-1 step -1 downto 1
```

```
            InvShiftRows(state)
            InvSubBytes(state)
            AddRoundKey(state, w[round*Nb, (round+1)*Nb-1])
            InvMixColumns(state)
      end for
      InvShiftRows(state)
      InvSubBytes(state)
      AddRoundKey(state, w[0, Nb-1])
      out = state
end
```

Key Expansion

```
KeyExpansion(byte key[4*Nk], word w[Nb*(Nr+1)], Nk)
begin
      word temp
      i = 0
      while (i < Nk)
            w[i] = word(key[4*i], key[4*i+1], key[4*i+2], key[4*i+3])
            i = i+1
      end while
      i = Nk
      while (i < Nb * (Nr+1)]
            temp = w[i-1]
            if (i mod Nk = 0)
                  temp = SubWord(RotWord(temp)) xor Rcon[i/Nk]
            else if (Nk > 6 and i mod Nk = 4)
                  temp = SubWord(temp)
            end if
            w[i] = w[i-Nk] xor temp
            i = i + 1
      end while
end
```

Tham khảo:

http://en.wikipedia.org/wiki/Advanced_Encryption_Standard
http://en.wikipedia.org/wiki/Rijndael_key_schedule