AES

Généré par Doxygen 1.9.7

Chapitre 1

Liste des bogues

Fichier aes.c
No known bugs.

Fichier aes.h
No known bugs.

Fichier bitmap.c
Fichiers illisibles sous Windows uniquement.

Fichier cipher.c
No known bugs.

Fichier cipher.h
No known bugs.

Fichier entropie.c
No known bugs.

Fichier entropie.h
No known bugs.

Fichier tests.c
No known bugs.

Fichier tools.c
No known bugs.

Fichier tools.h
No known bugs.

Chapitre 2

Index des fichiers

2.1 Liste des fichiers

Liste de tous les fichiers avec une brève description :

Chapitre 3

Documentation des fichiers

3.1 Référence du fichier c/aes.c

AES encryption and decryption protocol.

```
#include ¨cipher.h¨
#include ¨tools.h¨
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Fonctions

- byte ∗ keyprocess (char ∗key, int keysize, int ∗nr)
  Process of keyExpansion.
- char ∗ hextoascii (const char ∗in)
  Convert a hexadecimal string to an ascii string.
- char ∗ asciitohex (const char ∗in)
  Convert an ascii string to a hexadecimal string.
- int aes_encrypt (char ∗data, int size, char ∗key, int keysize, int cbc)
  Encrypt data with AES.
- int aes_decrypt (char ∗data, int size, char ∗key, int keysize, int cbc)
  Decrypt data with AES.

3.1.1 Description détaillée

AES encryption and decryption protocol.

Contient les fonctions de chiffrement et de déchiffrement AES pour des données de taille multiple de 16 octets.

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.1.2 Documentation des fonctions

3.1.2.1 aes_decrypt()

```
int aes_decrypt (
            char * data,
            int size,
            char * key,
            int keysize,
            int cbc )
```

Decrypt data with AES.

Paramètres

data	The data to decrypt
size	The size of the data (multiple of 16 bytes)
key	The key to decrypt the data
keysize	The size of the key (16, 24, 32 bytes)
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

0 if success, 1 if error

3.1.2.2 aes_encrypt()

```
int aes_encrypt (
            char * data,
            int size,
            char * key,
            int keysize,
            int cbc )
```

Encrypt data with AES.

Paramètres

data	The data to encrypt
size	The size of the data (multiple of 16 bytes)
key	The key to encrypt the data
keysize	The size of the key (16, 24, 32 bytes)
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

0 if success, 1 if error

3.1.2.3 asciitohex()

```
char * asciitohex (
             const char * in )
```

Convert an ascii string to a hexadecimal string.

Paramètres

in	The ascii string

Renvoie

The hexadecimal string

3.1.2.4 hextoascii()

```
char * hextoascii (
             const char * in )
```

Convert a hexadecimal string to an ascii string.

Paramètres

in	The hexadecimal string

Renvoie

The ascii string

3.1.2.5 keyprocess()

```
byte * keyprocess (
             char * key,
             int keysize,
             int * nr )
```

Process of keyExpansion.

Alloue la mémoire pour la clé étendue Calcule les valeurs de Nr et Nk Rempli la clé étendue

Paramètres

key	The initial key (16, 24, 32 bytes)
keysize	The size of the key (16, 24, 32 bytes)
nr	The number of rounds (10, 12, 14) (output variable)

Renvoie

The extended key

3.2 Référence du fichier c/aes.h

Function prototypes of the aes method.

Définitions de type

- typedef unsigned char byte

Fonctions

- byte ∗ keyprocess (char ∗key, int keysize, int ∗nr)
  Process of keyExpansion.
- char ∗ hextoascii (const char ∗in)
  Convert a hexadecimal string to an ascii string.
- char ∗ asciitohex (const char ∗in)
  Convert an ascii string to a hexadecimal string.
- int aes_encrypt (char ∗data, int size, char ∗key, int keysize, int cbc)
  Encrypt data with AES.
- int aes_decrypt (char ∗data, int size, char ∗key, int keysize, int cbc)
  Decrypt data with AES.

3.2.1 Description détaillée

Function prototypes of the aes method.

Contient les prototypes pour le protocole AES

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.2.2 Documentation des définitions de type

3.2.2.1 byte

```
typedef unsigned char byte
```

3.2.3 Documentation des fonctions

3.2.3.1 aes_decrypt()

```
int aes_decrypt (
             char * data,
             int size,
             char * key,
             int keysize,
             int cbc )
```

Decrypt data with AES.

Paramètres

data	The data to decrypt
size	The size of the data (multiple of 16 bytes)
key	The key to decrypt the data
keysize	The size of the key (16, 24, 32 bytes)
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

0 if success, 1 if error

3.2.3.2 aes_encrypt()

```
int aes_encrypt (
             char * data,
             int size,
             char * key,
             int keysize,
             int cbc )
```

Encrypt data with AES.

Paramètres

data	The data to encrypt
size	The size of the data (multiple of 16 bytes)
key	The key to encrypt the data
keysize	The size of the key (16, 24, 32 bytes)
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

0 if success, 1 if error

3.2.3.3 asciitohex()

```
char * asciitohex (
             const char * in )
```

Convert an ascii string to a hexadecimal string.

Paramètres

in	The ascii string

Renvoie

The hexadecimal string

3.2.3.4 hextoascii()

```
char * hextoascii (
             const char * in )
```

Convert a hexadecimal string to an ascii string.

Paramètres

in	The hexadecimal string

Renvoie

The ascii string

3.2.3.5 keyprocess()

```
byte * keyprocess (
             char * key,
             int keysize,
             int * nr )
```

Process of keyExpansion.

Alloue la mémoire pour la clé étendue Calcule les valeurs de Nr et Nk Rempli la clé étendue

Paramètres

key	The initial key (16, 24, 32 bytes)
keysize	The size of the key (16, 24, 32 bytes)
nr	The number of rounds (10, 12, 14) (output variable)

Renvoie

The extended key

3.3 aes.h

Aller à la documentation de ce fichier.

```

/* -- Defines -- */
typedef unsigned char byte;


/* -- Functions -- */
byte *keyprocess(char *key, int keysize, int *nr);

char* hextoascii(const char* in);
```

```

char* asciitohex(const char* in);

int aes_encrypt (char *data, int size, char *key, int keysize, int cbc);

int aes_decrypt (char *data, int size, char *key, int keysize, int cbc);
```

3.4 Référence du fichier c/bitmap.c

BMP encryption and decryption.

```
#include "cipher.h"
#include "tools.h"
#include "aes.h"
#include "entropie.h"
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include <time.h>
```

Fonctions

- void ecrireBMP (char *filename, unsigned char *info, unsigned char *data, int size)
  Create a BMP file.
- void chiffrerBMP (char *filename, char *output_name, int cbc)
  Encrypt a BMP file.
- void dechiffrerBMP (char *filename, char *output_name, int cbc)
  Decrypt a BMP file.
- int main (int argc, char **argv)

3.4.1 Description détaillée

BMP encryption and decryption.

Contient les fonctions de chiffrement et de déchiffrement pour des fichiers BMP.

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue Fichiers illisibles sous Windows uniquement.

3.4.2 Documentation des fonctions

3.4.2.1 chiffrerBMP()

```
void chiffrerBMP (
            char * filename,
            char * output_name,
            int cbc )
```

Encrypt a BMP file.

Paramètres

filename	Filename of the input file
output_name	Filename of the output file
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

Void

3.4.2.2 dechiffrerBMP()

```
void dechiffrerBMP (
            char * filename,
            char * output_name,
            int cbc )
```

Decrypt a BMP file.

Paramètres

filename	Filename of the input file
output_name	Filename of the output file
cbc	Enable the CBC mode (1 enabled and 0 for ECB mode)

Renvoie

Void

3.4.2.3 ecrireBMP()

```
void ecrireBMP (
            char * filename,
            unsigned char * info,
            unsigned char * data,
            int size )
```

Create a BMP file.

Paramètres

filename	Filename of the output file
info	The header of the BMP file
data	The data of the BMP file
size	The size of the data

Renvoie

Void

3.4.2.4 main()

```
int main (
            int argc,
            char ** argv )
```

3.5 Référence du fichier c/cipher.c

Cipher method.

```
#include ¨tools.h¨
#include <string.h>
```

Fonctions

- void addRoundKey (byte state[ ], byte w[ ], int round)

  Add the key to the state (xor operation)
- void subBytes (byte state[ ], const byte box[256], int length)

  Substitute the bytes of the state with a box.
- void shiftOneRow (byte state[ ], int row, int direction, int shift)

  Shift one row of the state.
- void shiftRows (byte state[ ])

  Shift all the rows of the state.
- void invShiftRows (byte state[ ])

  Inverse process of shiftRows.
- byte multiTab (byte a, byte b)
- void mixColumns (byte state[ ], const int inv)

  Mix the columns of the state.
- void rotWord (byte state[4])

  1 byte rigth rotation of a 4 byte state
- void rcon (int i, byte out[4])

  Create the rcon polynome associated to the round.
- void keyExpansion (byte key[ ], byte w[ ], int nk, int nr)

  Key expansion method.
- void cipher (byte in[ ], byte w[ ], int nr)

  Cipher method.
- void invCipher (byte in[ ], byte w[ ], int nr)

  Inverse cipher method.

Variables

- const byte sbox [256] = {0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76, 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, 0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15, 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, 0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84, 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, 0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, 0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73, 0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, 0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79, 0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08, 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a, 0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e, 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf, 0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16}

- const byte invSbox [256] = {0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40, 0xa3, 0x9e, 0x81, 0xf3, 0xd7, 0xfb, 0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87, 0x34, 0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb, 0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d, 0xee, 0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e, 0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24, 0xb2, 0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25, 0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92, 0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed, 0xb9, 0xda, 0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84, 0x90, 0xd8, 0xab, 0x00, 0x8c, 0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06, 0xd0, 0x2c, 0x1e, 0x8f, 0xca, 0x3f, 0x0f, 0x02, 0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b, 0x3a, 0x91, 0x11, 0x41, 0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73, 0x96, 0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e, 0x47, 0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b, 0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd, 0x5a, 0xf4, 0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xec, 0x5f, 0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f, 0x93, 0xc9, 0x9c, 0xef, 0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0, 0xc8, 0xeb, 0xbb, 0x3c, 0x83, 0x53, 0x99, 0x61, 0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0c, 0x7d}
- const byte a_x_mixColumns [16] = {0x02, 0x03, 0x01, 0x01, 0x01, 0x02, 0x03, 0x01, 0x01, 0x01, 0x02, 0x03, 0x03, 0x01, 0x01, 0x02}
- const byte a_x_invMixColumns [16] = {0x0e, 0x0b, 0x0d, 0x09, 0x09, 0x0e, 0x0b, 0x0d, 0x0d, 0x09, 0x0e, 0x0b, 0x0b, 0x0d, 0x09, 0x0e}
- const byte table02 [256] = {0x00, 0x02, 0x04, 0x06, 0x08, 0x0a, 0x0c, 0x0e, 0x10, 0x12, 0x14, 0x16, 0x18, 0x1a, 0x1c, 0x1e, 0x20, 0x22, 0x24, 0x26, 0x28, 0x2a, 0x2c, 0x2e, 0x30, 0x32, 0x34, 0x36, 0x38, 0x3a, 0x3c, 0x3e, 0x40, 0x42, 0x44, 0x46, 0x48, 0x4a, 0x4c, 0x4e, 0x50, 0x52, 0x54, 0x56, 0x58, 0x5a, 0x5c, 0x5e, 0x60, 0x62, 0x64, 0x66, 0x68, 0x6a, 0x6c, 0x6e, 0x70, 0x72, 0x74, 0x76, 0x78, 0x7a, 0x7c, 0x7e, 0x80, 0x82, 0x84, 0x86, 0x88, 0x8a, 0x8c, 0x8e, 0x90, 0x92, 0x94, 0x96, 0x98, 0x9a, 0x9c, 0x9e, 0xa0, 0xa2, 0xa4, 0xa6, 0xa8, 0xaa, 0xac, 0xae, 0xb0, 0xb2, 0xb4, 0xb6, 0xb8, 0xba, 0xbc, 0xbe, 0xc0, 0xc2, 0xc4, 0xc6, 0xc8, 0xca, 0xcc, 0xce, 0xd0, 0xd2, 0xd4, 0xd6, 0xd8, 0xda, 0xdc, 0xde, 0xe0, 0xe2, 0xe4, 0xe6, 0xe8, 0xea, 0xec, 0xee, 0xf0, 0xf2, 0xf4, 0xf6, 0xf8, 0xfa, 0xfc, 0xfe, 0x1b, 0x19, 0x1f, 0x1d, 0x13, 0x11, 0x17, 0x15, 0x0b, 0x09, 0x0f, 0x0d, 0x03, 0x01, 0x07, 0x05, 0x3b, 0x39, 0x3f, 0x3d, 0x33, 0x31, 0x37, 0x35, 0x2b, 0x29, 0x2f, 0x2d, 0x23, 0x21, 0x27, 0x25, 0x5b, 0x59, 0x5f, 0x5d, 0x53, 0x51, 0x57, 0x55, 0x4b, 0x49, 0x4f, 0x4d, 0x43, 0x41, 0x47, 0x45, 0x7b, 0x79, 0x7f, 0x7d, 0x73, 0x71, 0x77, 0x75, 0x6b, 0x69, 0x6f, 0x6d, 0x63, 0x61, 0x67, 0x65, 0x9b, 0x99, 0x9f, 0x9d, 0x93, 0x91, 0x97, 0x95, 0x8b, 0x89, 0x8f, 0x8d, 0x83, 0x81, 0x87, 0x85, 0xbb, 0xb9, 0xbf, 0xbd, 0xb3, 0xb1, 0xb7, 0xb5, 0xab, 0xa9, 0xaf, 0xad, 0xa3, 0xa1, 0xa7, 0xa5, 0xdb, 0xd9, 0xdf, 0xdd, 0xd3, 0xd1, 0xd7, 0xd5, 0xcb, 0xc9, 0xcf, 0xcd, 0xc3, 0xc1, 0xc7, 0xc5, 0xfb, 0xf9, 0xff, 0xfd, 0xf3, 0xf1, 0xf7, 0xf5, 0xeb, 0xe9, 0xef, 0xed, 0xe3, 0xe1, 0xe7, 0xe5}
- const byte table03 [256] = {0x00, 0x03, 0x06, 0x05, 0x0c, 0x0f, 0x0a, 0x09, 0x18, 0x1b, 0x1e, 0x1d, 0x14, 0x17, 0x12, 0x11, 0x30, 0x33, 0x36, 0x35, 0x3c, 0x3f, 0x3a, 0x39, 0x28, 0x2b, 0x2e, 0x2d, 0x24, 0x27, 0x22, 0x21, 0x60, 0x63, 0x66, 0x65, 0x6c, 0x6f, 0x6a, 0x69, 0x78, 0x7b, 0x7e, 0x7d, 0x74, 0x77, 0x72, 0x71, 0x50, 0x53, 0x56, 0x55, 0x5c, 0x5f, 0x5a, 0x59, 0x48, 0x4b, 0x4e, 0x4d, 0x44, 0x47, 0x42, 0x41, 0xc0, 0xc3, 0xc6, 0xc5, 0xcc, 0xcf, 0xca, 0xc9, 0xd8, 0xdb, 0xde, 0xdd, 0xd4, 0xd7, 0xd2, 0xd1, 0xf0, 0xf3, 0xf6, 0xf5, 0xfc, 0xff, 0xfa, 0xf9, 0xe8, 0xeb, 0xee, 0xed, 0xe4, 0xe7, 0xe2, 0xe1, 0xa0, 0xa3, 0xa6, 0xa5, 0xac, 0xaf, 0xaa, 0xa9, 0xb8, 0xbb, 0xbe, 0xbd, 0xb4, 0xb7, 0xb2, 0xb1, 0x90, 0x93, 0x96, 0x95, 0x9c, 0x9f, 0x9a, 0x99, 0x88, 0x8b, 0x8e, 0x8d, 0x84, 0x87, 0x82, 0x81, 0x9b, 0x98, 0x9d, 0x9e, 0x97, 0x94, 0x91, 0x92, 0x83, 0x80, 0x85, 0x86, 0x8f, 0x8c, 0x89, 0x8a, 0xab, 0xa8, 0xad, 0xae, 0xa7, 0xa4, 0xa1, 0xa2, 0xb3, 0xb0, 0xb5, 0xb6, 0xbf, 0xbc, 0xb9, 0xba, 0xfb, 0xf8, 0xfd, 0xfe, 0xf7, 0xf4, 0xf1, 0xf2, 0xe3, 0xe0, 0xe5, 0xe6, 0xef, 0xec, 0xe9, 0xea, 0xcb, 0xc8, 0xcd, 0xce, 0xc7, 0xc4, 0xc1, 0xc2, 0xd3, 0xd0, 0xd5, 0xd6, 0xdf, 0xdc, 0xd9, 0xda, 0x5b, 0x58, 0x5d, 0x5e, 0x57, 0x54, 0x51, 0x52, 0x43, 0x40, 0x45, 0x46, 0x4f, 0x4c, 0x49, 0x4a, 0x6b, 0x68, 0x6d, 0x6e, 0x67, 0x64, 0x61, 0x62, 0x73, 0x70, 0x75, 0x76, 0x7f, 0x7c, 0x79, 0x7a, 0x3b, 0x38, 0x3d, 0x3e, 0x37, 0x34, 0x31, 0x32, 0x23, 0x20, 0x25, 0x26, 0x2f, 0x2c, 0x29, 0x2a, 0x0b, 0x08, 0x0d, 0x0e, 0x07, 0x04, 0x01, 0x02, 0x13, 0x10, 0x15, 0x16, 0x1f, 0x1c, 0x19, 0x1a}
- const byte table09 [256] = {0x00, 0x09, 0x12, 0x1b, 0x24, 0x2d, 0x36, 0x3f, 0x48, 0x41, 0x5a, 0x53, 0x6c, 0x65, 0x7e, 0x77, 0x90, 0x99, 0x82, 0x8b, 0xb4, 0xbd, 0xa6, 0xaf, 0xd8, 0xd1, 0xca, 0xc3, 0xfc, 0xf5, 0xee, 0xe7, 0x3b, 0x32, 0x29, 0x20, 0x1f, 0x16, 0x0d, 0x04, 0x73, 0x7a, 0x61, 0x68, 0x57, 0x5e, 0x45, 0x4c, 0xab, 0xa2, 0xb9, 0xb0, 0x8f, 0x86, 0x9d, 0x94, 0xe3, 0xea, 0xf1, 0xf8, 0xc7, 0xce, 0xd5, 0xdc, 0x76, 0x7f, 0x64, 0x6d, 0x52, 0x5b, 0x40, 0x49, 0x3e, 0x37, 0x2c, 0x25, 0x1a, 0x13, 0x08, 0x01, 0xe6, 0xef, 0xf4, 0xfd, 0xc2, 0xcb, 0xd0, 0xd9, 0xae, 0xa7, 0xbc, 0xb5, 0x8a, 0x83, 0x98, 0x91, 0x4d, 0x44, 0x5f, 0x56, 0x69, 0x60, 0x7b, 0x72, 0x05, 0x0c, 0x17, 0x1e, 0x21, 0x28, 0x33, 0x3a, 0xdd, 0xd4, 0xcf, 0xc6, 0xf9, 0xf0, 0xeb, 0xe2, 0x95, 0x9c, 0x87, 0x8e, 0xb1, 0xb8, 0xa3, 0xaa, 0xec, 0xe5, 0xfe, 0xf7, 0xc8, 0xc1, 0xda, 0xd3, 0xa4, 0xad, 0xb6, 0xbf, 0x80, 0x89, 0x92, 0x9b, 0x7c, 0x75, 0x6e, 0x67, 0x58, 0x51, 0x4a, 0x43, 0x34, 0x3d, 0x26, 0x2f, 0x10, 0x19, 0x02, 0x0b, 0xd7, 0xde, 0xc5, 0xcc, 0xf3, 0xfa, 0xe1, 0xe8, 0x9f, 0x96, 0x8d, 0x84, 0xbb, 0xb2, 0xa9,

0xa0, 0x47, 0x4e, 0x55, 0x5c, 0x63, 0x6a, 0x71, 0x78, 0x0f, 0x06, 0x1d, 0x14, 0x2b, 0x22, 0x39, 0x30, 0x9a, 0x93, 0x88, 0x81, 0xbe, 0xb7, 0xac, 0xa5, 0xd2, 0xdb, 0xc0, 0xc9, 0xf6, 0xff, 0xe4, 0xed, 0x0a, 0x03, 0x18, 0x11, 0x2e, 0x27, 0x3c, 0x35, 0x42, 0x4b, 0x50, 0x59, 0x66, 0x6f, 0x74, 0x7d, 0xa1, 0xa8, 0xb3, 0xba, 0x85, 0x8c, 0x97, 0x9e, 0xe9, 0xe0, 0xfb, 0xf2, 0xcd, 0xc4, 0xdf, 0xd6, 0x31, 0x38, 0x23, 0x2a, 0x15, 0x1c, 0x07, 0x0e, 0x79, 0x70, 0x6b, 0x62, 0x5d, 0x54, 0x4f, 0x46}

- const byte table0b [256] = {0x00, 0x0b, 0x16, 0x1d, 0x2c, 0x27, 0x3a, 0x31, 0x58, 0x53, 0x4e, 0x45, 0x74, 0x7f, 0x62, 0x69, 0xb0, 0xbb, 0xa6, 0xad, 0x9c, 0x97, 0x8a, 0x81, 0xe8, 0xe3, 0xfe, 0xf5, 0xc4, 0xcf, 0xd2, 0xd9, 0x7b, 0x70, 0x6d, 0x66, 0x57, 0x5c, 0x41, 0x4a, 0x23, 0x28, 0x35, 0x3e, 0x0f, 0x04, 0x19, 0x12, 0xcb, 0xc0, 0xdd, 0xd6, 0xe7, 0xec, 0xf1, 0xfa, 0x93, 0x98, 0x85, 0x8e, 0xbf, 0xb4, 0xa9, 0xa2, 0xf6, 0xfd, 0xe0, 0xeb, 0xda, 0xd1, 0xcc, 0xc7, 0xae, 0xa5, 0xb8, 0xb3, 0x82, 0x89, 0x94, 0x9f, 0x46, 0x4d, 0x50, 0x5b, 0x6a, 0x61, 0x7c, 0x77, 0x1e, 0x15, 0x08, 0x03, 0x32, 0x39, 0x24, 0x2f, 0x8d, 0x86, 0x9b, 0x90, 0xa1, 0xaa, 0xb7, 0xbc, 0xd5, 0xde, 0xc3, 0xc8, 0xf9, 0xf2, 0xef, 0xe4, 0x3d, 0x36, 0x2b, 0x20, 0x11, 0x1a, 0x07, 0x0c, 0x65, 0x6e, 0x73, 0x78, 0x49, 0x42, 0x5f, 0x54, 0xf7, 0xfc, 0xe1, 0xea, 0xdb, 0xd0, 0xcd, 0xc6, 0xaf, 0xa4, 0xb9, 0xb2, 0x83, 0x88, 0x95, 0x9e, 0x47, 0x4c, 0x51, 0x5a, 0x6b, 0x60, 0x7d, 0x76, 0x1f, 0x14, 0x09, 0x02, 0x33, 0x38, 0x25, 0x2e, 0x8c, 0x87, 0x9a, 0x91, 0xa0, 0xab, 0xb6, 0xbd, 0xd4, 0xdf, 0xc2, 0xc9, 0xf8, 0xf3, 0xee, 0xe5, 0x3c, 0x37, 0x2a, 0x21, 0x10, 0x1b, 0x06, 0x0d, 0x64, 0x6f, 0x72, 0x79, 0x48, 0x43, 0x5e, 0x55, 0x01, 0x0a, 0x17, 0x1c, 0x2d, 0x26, 0x3b, 0x30, 0x59, 0x52, 0x4f, 0x44, 0x75, 0x7e, 0x63, 0x68, 0xb1, 0xba, 0xa7, 0xac, 0x9d, 0x96, 0x8b, 0x80, 0xe9, 0xe2, 0xff, 0xf4, 0xc5, 0xce, 0xd3, 0xd8, 0x7a, 0x71, 0x6c, 0x67, 0x56, 0x5d, 0x40, 0x4b, 0x22, 0x29, 0x34, 0x3f, 0x0e, 0x05, 0x18, 0x13, 0xca, 0xc1, 0xdc, 0xd7, 0xe6, 0xed, 0xf0, 0xfb, 0x92, 0x99, 0x84, 0x8f, 0xbe, 0xb5, 0xa8, 0xa3}
- const byte table0d [256] = {0x00, 0x0d, 0x1a, 0x17, 0x34, 0x39, 0x2e, 0x23, 0x68, 0x65, 0x72, 0x7f, 0x5c, 0x51, 0x46, 0x4b, 0xd0, 0xdd, 0xca, 0xc7, 0xe4, 0xe9, 0xfe, 0xf3, 0xb8, 0xb5, 0xa2, 0xaf, 0x8c, 0x81, 0x96, 0x9b, 0xbb, 0xb6, 0xa1, 0xac, 0x8f, 0x82, 0x95, 0x98, 0xd3, 0xde, 0xc9, 0xc4, 0xe7, 0xea, 0xfd, 0xf0, 0x6b, 0x66, 0x71, 0x7c, 0x5f, 0x52, 0x45, 0x48, 0x03, 0x0e, 0x19, 0x14, 0x37, 0x3a, 0x2d, 0x20, 0x6d, 0x60, 0x77, 0x7a, 0x59, 0x54, 0x43, 0x4e, 0x05, 0x08, 0x1f, 0x12, 0x31, 0x3c, 0x2b, 0x26, 0xbd, 0xb0, 0xa7, 0xaa, 0x89, 0x84, 0x93, 0x9e, 0xd5, 0xd8, 0xcf, 0xc2, 0xe1, 0xec, 0xfb, 0xf6, 0xd6, 0xdb, 0xcc, 0xc1, 0xe2, 0xef, 0xf8, 0xf5, 0xbe, 0xb3, 0xa4, 0xa9, 0x8a, 0x87, 0x90, 0x9d, 0x06, 0x0b, 0x1c, 0x11, 0x32, 0x3f, 0x28, 0x25, 0x6e, 0x63, 0x74, 0x79, 0x5a, 0x57, 0x40, 0x4d, 0xda, 0xd7, 0xc0, 0xcd, 0xee, 0xe3, 0xf4, 0xf9, 0xb2, 0xbf, 0xa8, 0xa5, 0x86, 0x8b, 0x9c, 0x91, 0x0a, 0x07, 0x10, 0x1d, 0x3e, 0x33, 0x24, 0x29, 0x62, 0x6f, 0x78, 0x75, 0x56, 0x5b, 0x4c, 0x41, 0x61, 0x6c, 0x7b, 0x76, 0x55, 0x58, 0x4f, 0x42, 0x09, 0x04, 0x13, 0x1e, 0x3d, 0x30, 0x27, 0x2a, 0xb1, 0xbc, 0xab, 0xa6, 0x85, 0x88, 0x9f, 0x92, 0xd9, 0xd4, 0xc3, 0xce, 0xed, 0xe0, 0xf7, 0xfa, 0xb7, 0xba, 0xad, 0xa0, 0x83, 0x8e, 0x99, 0x94, 0xdf, 0xd2, 0xc5, 0xc8, 0xeb, 0xe6, 0xf1, 0xfc, 0x67, 0x6a, 0x7d, 0x70, 0x53, 0x5e, 0x49, 0x44, 0x0f, 0x02, 0x15, 0x18, 0x3b, 0x36, 0x21, 0x2c, 0x0c, 0x01, 0x16, 0x1b, 0x38, 0x35, 0x22, 0x2f, 0x64, 0x69, 0x7e, 0x73, 0x50, 0x5d, 0x4a, 0x47, 0xdc, 0xd1, 0xc6, 0xcb, 0xe8, 0xe5, 0xf2, 0xff, 0xb4, 0xb9, 0xae, 0xa3, 0x80, 0x8d, 0x9a, 0x97}
- const byte table0e [256] = {0x00, 0x0e, 0x1c, 0x12, 0x38, 0x36, 0x24, 0x2a, 0x70, 0x7e, 0x6c, 0x62, 0x48, 0x46, 0x54, 0x5a, 0xe0, 0xee, 0xfc, 0xf2, 0xd8, 0xd6, 0xc4, 0xca, 0x90, 0x9e, 0x8c, 0x82, 0xa8, 0xa6, 0xb4, 0xba, 0xdb, 0xd5, 0xc7, 0xc9, 0xe3, 0xed, 0xff, 0xf1, 0xab, 0xa5, 0xb7, 0xb9, 0x93, 0x9d, 0x8f, 0x81, 0x3b, 0x35, 0x27, 0x29, 0x03, 0x0d, 0x1f, 0x11, 0x4b, 0x45, 0x57, 0x59, 0x73, 0x7d, 0x6f, 0x61, 0xad, 0xa3, 0xb1, 0xbf, 0x95, 0x9b, 0x89, 0x87, 0xdd, 0xd3, 0xc1, 0xcf, 0xe5, 0xeb, 0xf9, 0xf7, 0x4d, 0x43, 0x51, 0x5f, 0x75, 0x7b, 0x69, 0x67, 0x3d, 0x33, 0x21, 0x2f, 0x05, 0x0b, 0x19, 0x17, 0x76, 0x78, 0x6a, 0x64, 0x4e, 0x40, 0x52, 0x5c, 0x06, 0x08, 0x1a, 0x14, 0x3e, 0x30, 0x22, 0x2c, 0x96, 0x98, 0x8a, 0x84, 0xae, 0xa0, 0xb2, 0xbc, 0xe6, 0xe8, 0xfa, 0xf4, 0xde, 0xd0, 0xc2, 0xcc, 0x41, 0x4f, 0x5d, 0x53, 0x79, 0x77, 0x65, 0x6b, 0x31, 0x3f, 0x2d, 0x23, 0x09, 0x07, 0x15, 0x1b, 0xa1, 0xaf, 0xbd, 0xb3, 0x99, 0x97, 0x85, 0x8b, 0xd1, 0xdf, 0xcd, 0xc3, 0xe9, 0xe7, 0xf5, 0xfb, 0x9a, 0x94, 0x86, 0x88, 0xa2, 0xac, 0xbe, 0xb0, 0xea, 0xe4, 0xf6, 0xf8, 0xd2, 0xdc, 0xce, 0xc0, 0x7a, 0x74, 0x66, 0x68, 0x42, 0x4c, 0x5e, 0x50, 0x0a, 0x04, 0x16, 0x18, 0x32, 0x3c, 0x2e, 0x20, 0xec, 0xe2, 0xf0, 0xfe, 0xd4, 0xda, 0xc8, 0xc6, 0x9c, 0x92, 0x80, 0x8e, 0xa4, 0xaa, 0xb8, 0xb6, 0x0c, 0x02, 0x10, 0x1e, 0x34, 0x3a, 0x28, 0x26, 0x7c, 0x72, 0x60, 0x6e, 0x44, 0x4a, 0x58, 0x56, 0x37, 0x39, 0x2b, 0x25, 0x0f, 0x01, 0x13, 0x1d, 0x47, 0x49, 0x5b, 0x55, 0x7f, 0x71, 0x63, 0x6d, 0xd7, 0xd9, 0xcb, 0xc5, 0xef, 0xe1, 0xf3, 0xfd, 0xa7, 0xa9, 0xbb, 0xb5, 0x9f, 0x91, 0x83, 0x8d}

3.5.1 Description détaillée

Cipher method.

Contient tous les fonctions nécessaires au chiffrement (et au déchiffrement) d'un bloc de 16 octets avec une clé.

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.5.2 Documentation des fonctions

3.5.2.1 addRoundKey()

```
void addRoundKey (
            byte state[ ],
            byte w[ ],
            int round )
```

Add the key to the state (xor operation)

Paramètres

state	The current state (16 bytes)
w	The entire key
round	The current round (relative to Nr)

Renvoie

Void

3.5.2.2 cipher()

```
void cipher (
            byte in[ ],
            byte w[ ],
            int nr )
```

Cipher method.

Paramètres

in	The input block (16 bytes) enlarged over the rounds
w	The expanded key (16∗(Nr+1) bytes)
nr	The number of rounds

Renvoie

Void

3.5.2.3 invCipher()

```
void invCipher (
             byte in[ ],
             byte w[ ],
             int nr )
```

Inverse cipher method.

Paramètres

in	The input block (16 bytes) enlarged over the rounds
w	The expanded key (16∗(Nr+1) bytes)
nr	The number of rounds

Renvoie

Void

3.5.2.4 invShiftRows()

```
void invShiftRows (
             byte state[ ] )
```

Inverse process of shiftRows.

Paramètres

state	The current state (16 bytes)

Renvoie

Void

3.5.2.5 keyExpansion()

```
void keyExpansion (
             byte key[ ],
             byte w[ ],
             int nk,
             int nr )
```

Key expansion method.

Paramètres

key	The key (16, 24 or 32 bytes)
w	The expanded key generated (16∗(Nr+1) bytes)
nk	The number of words in the key (4, 6 or 8 refering to the key size (16, 24 or 32 bytes)))
nr	The number of rounds

Renvoie

Void

3.5.2.6 mixColumns()

```
void mixColumns (
             byte state[],
             const int inv )
```

Mix the columns of the state.

Paramètres

state	The current state (16 bytes)
inv	1 for the Inverse Mix Columns, 0 for the Mix Columns

Renvoie

Void

3.5.2.7 multiTab()

```
byte multiTab (
             byte a,
             byte b )
```

3.5.2.8 rcon()

```
void rcon (
             int i,
             byte out[4] )
```

Create the rcon polynome associated to the round.

Paramètres

i	The current round
out	The word generated (4 bytes)

Renvoie

Void

3.5.2.9 rotWord()

```
void rotWord (
             byte state[4] )
```

1 byte rigth rotation of a 4 byte state

Paramètres

state	The current word (4 bytes)

Renvoie

Void

3.5.2.10 shiftOneRow()

```
void shiftOneRow (
            byte state[],
            int row,
            int direction,
            int shift )
```

Shift one row of the state.

Paramètres

state	The current state (16 bytes)
row	The row to shift
direction	The direction of the shift (1 for right, -1 for left)
shift	The number of shifts

Renvoie

Void

3.5.2.11 shiftRows()

```
void shiftRows (
            byte state[] )
```

Shift all the rows of the state.

Paramètres

state	The current state (16 bytes)

Renvoie

Void

3.5.2.12 subBytes()

```
void subBytes (
            byte state[],
            const byte box[256],
            int length )
```

Substitute the bytes of the state with a box.

Paramètres

state	The current state (16 bytes)
box	Either the S-Box or the inverse S-Box (256 bytes)
length	The length of the state (16 for subBytes and 4 for subWord)

Renvoie

Void

3.5.3 Documentation des variables

3.5.3.1 a_x_invMixColumns

```
const byte a_x_invMixColumns[16] = {0x0e, 0x0b, 0x0d, 0x09, 0x09, 0x0e, 0x0b, 0x0d, 0x0d,
0x09, 0x0e, 0x0b, 0x0b, 0x0d, 0x09, 0x0e}
```

3.5.3.2 a_x_mixColumns

```
const byte a_x_mixColumns[16] = {0x02, 0x03, 0x01, 0x01, 0x01, 0x02, 0x03, 0x01, 0x01, 0x01,
0x02, 0x03, 0x03, 0x01, 0x01, 0x02}
```

3.5.3.3 invSbox

```
const byte invSbox[256] = {0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40, 0xa3,
0x9e, 0x81, 0xf3, 0xd7, 0xfb, 0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87, 0x34, 0x8e,
0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb, 0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d, 0xee,
0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e, 0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24, 0xb2,
0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25, 0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68, 0x98,
0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92, 0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed,
0xb9, 0xda, 0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84, 0x90, 0xd8, 0xab, 0x00, 0x8c,
0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06, 0xd0, 0x2c, 0x1e, 0x8f,
0xca, 0x3f, 0x0f, 0x02, 0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b, 0x3a, 0x91, 0x11,
0x41, 0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73, 0x96, 0xac,
0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e, 0x47,
0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b,
0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd, 0x5a,
0xf4, 0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27, 0x80,
0xec, 0x5f, 0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f, 0x93,
0xc9, 0x9c, 0xef, 0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0, 0xc8, 0xeb, 0xbb, 0x3c,
0x83, 0x53, 0x99, 0x61, 0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69, 0x14,
0x63, 0x55, 0x21, 0x0c, 0x7d}
```

3.5.3.4 sbox

```
const byte sbox[256] = {0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67,
0x2b, 0xfe, 0xd7, 0xab, 0x76, 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4,
0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, 0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34,
0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15, 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a,
0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, 0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a,
0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84, 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc,
0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, 0xd0, 0xef, 0xaa, 0xfb, 0x43,
0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0x51, 0xa3, 0x40, 0x8f,
0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, 0xcd, 0x0c, 0x13,
0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73, 0x60, 0x81,
0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, 0xe0,
0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae,
0x08, 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd,
0x8b, 0x8a, 0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86,
0xc1, 0x1d, 0x9e, 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9,
0xce, 0x55, 0x28, 0xdf, 0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d,
0x0f, 0xb0, 0x54, 0xbb, 0x16}
```

3.5.3.5 table02

```
const byte table02[256] = {0x00, 0x02, 0x04, 0x06, 0x08, 0x0a, 0x0c, 0x0e, 0x10, 0x12, 0x14,
0x16, 0x18, 0x1a, 0x1c, 0x1e, 0x20, 0x22, 0x24, 0x26, 0x28, 0x2a, 0x2c, 0x2e, 0x30, 0x32,
0x34, 0x36, 0x38, 0x3a, 0x3c, 0x3e, 0x40, 0x42, 0x44, 0x46, 0x48, 0x4a, 0x4c, 0x4e, 0x50,
0x52, 0x54, 0x56, 0x58, 0x5a, 0x5c, 0x5e, 0x60, 0x62, 0x64, 0x66, 0x68, 0x6a, 0x6c, 0x6e,
0x70, 0x72, 0x74, 0x76, 0x78, 0x7a, 0x7c, 0x7e, 0x80, 0x82, 0x84, 0x86, 0x88, 0x8a, 0x8c,
0x8e, 0x90, 0x92, 0x94, 0x96, 0x98, 0x9a, 0x9c, 0x9e, 0xa0, 0xa2, 0xa4, 0xa6, 0xa8, 0xaa,
0xac, 0xae, 0xb0, 0xb2, 0xb4, 0xb6, 0xb8, 0xba, 0xbc, 0xbe, 0xc0, 0xc2, 0xc4, 0xc6, 0xc8,
0xca, 0xcc, 0xce, 0xd0, 0xd2, 0xd4, 0xd6, 0xd8, 0xda, 0xdc, 0xde, 0xe0, 0xe2, 0xe4, 0xe6,
0xe8, 0xea, 0xec, 0xee, 0xf0, 0xf2, 0xf4, 0xf6, 0xf8, 0xfa, 0xfc, 0xfe, 0x1b, 0x19, 0x1f,
0x1d, 0x13, 0x11, 0x17, 0x15, 0x0b, 0x09, 0x0f, 0x0d, 0x03, 0x01, 0x07, 0x05, 0x3b, 0x39,
0x3f, 0x3d, 0x33, 0x31, 0x37, 0x35, 0x2b, 0x29, 0x2f, 0x2d, 0x23, 0x21, 0x27, 0x25, 0x5b,
0x59, 0x5f, 0x5d, 0x53, 0x51, 0x57, 0x55, 0x4b, 0x49, 0x4f, 0x4d, 0x43, 0x41, 0x47, 0x45,
0x7b, 0x79, 0x7f, 0x7d, 0x73, 0x71, 0x77, 0x75, 0x6b, 0x69, 0x6f, 0x6d, 0x63, 0x61, 0x67,
0x65, 0x9b, 0x99, 0x9f, 0x9d, 0x93, 0x91, 0x97, 0x95, 0x8b, 0x89, 0x8f, 0x8d, 0x83, 0x81,
0x87, 0x85, 0xbb, 0xb9, 0xbf, 0xbd, 0xb3, 0xb1, 0xb7, 0xb5, 0xab, 0xa9, 0xaf, 0xad, 0xa3,
0xa1, 0xa7, 0xa5, 0xdb, 0xd9, 0xdf, 0xdd, 0xd3, 0xd1, 0xd7, 0xd5, 0xcb, 0xc9, 0xcf, 0xcd,
0xc3, 0xc1, 0xc7, 0xc5, 0xfb, 0xf9, 0xff, 0xfd, 0xf3, 0xf1, 0xf7, 0xf5, 0xeb, 0xe9, 0xef,
0xed, 0xe3, 0xe1, 0xe7, 0xe5}
```

3.5.3.6 table03

```
const byte table03[256] = {0x00, 0x03, 0x06, 0x05, 0x0c, 0x0f, 0x0a, 0x09, 0x18, 0x1b, 0x1e,
0x1d, 0x14, 0x17, 0x12, 0x11, 0x30, 0x33, 0x36, 0x35, 0x3c, 0x3f, 0x3a, 0x39, 0x28, 0x2b,
0x2e, 0x2d, 0x24, 0x27, 0x22, 0x21, 0x60, 0x63, 0x66, 0x65, 0x6c, 0x6f, 0x6a, 0x69, 0x78,
0x7b, 0x7e, 0x7d, 0x74, 0x77, 0x72, 0x71, 0x50, 0x53, 0x56, 0x55, 0x5c, 0x5f, 0x5a, 0x59,
0x48, 0x4b, 0x4e, 0x4d, 0x44, 0x47, 0x42, 0x41, 0xc0, 0xc3, 0xc6, 0xc5, 0xcc, 0xcf, 0xca,
0xc9, 0xd8, 0xdb, 0xde, 0xdd, 0xd4, 0xd7, 0xd2, 0xd1, 0xf0, 0xf3, 0xf6, 0xf5, 0xfc, 0xff,
0xfa, 0xf9, 0xe8, 0xeb, 0xee, 0xed, 0xe4, 0xe7, 0xe2, 0xe1, 0xa0, 0xa3, 0xa6, 0xa5, 0xac,
0xaf, 0xaa, 0xa9, 0xb8, 0xbb, 0xbe, 0xbd, 0xb4, 0xb7, 0xb2, 0xb1, 0x90, 0x93, 0x96, 0x95,
0x9c, 0x9f, 0x9a, 0x99, 0x88, 0x8b, 0x8e, 0x8d, 0x84, 0x87, 0x82, 0x81, 0x9b, 0x98, 0x9d,
0x9e, 0x97, 0x94, 0x91, 0x92, 0x83, 0x80, 0x85, 0x86, 0x8f, 0x8c, 0x89, 0x8a, 0xab, 0xa8,
0xad, 0xae, 0xa7, 0xa4, 0xa1, 0xa2, 0xb3, 0xb0, 0xb5, 0xb6, 0xbf, 0xbc, 0xb9, 0xba, 0xfb,
0xf8, 0xfd, 0xfe, 0xf7, 0xf4, 0xf1, 0xf2, 0xe3, 0xe0, 0xe5, 0xe6, 0xef, 0xec, 0xe9, 0xea,
0xcb, 0xc8, 0xcd, 0xce, 0xc7, 0xc4, 0xc1, 0xc2, 0xd3, 0xd0, 0xd5, 0xd6, 0xdf, 0xdc, 0xd9,
0xda, 0x5b, 0x58, 0x5d, 0x5e, 0x57, 0x54, 0x51, 0x52, 0x43, 0x40, 0x45, 0x46, 0x4f, 0x4c,
0x49, 0x4a, 0x6b, 0x68, 0x6d, 0x6e, 0x67, 0x64, 0x61, 0x62, 0x73, 0x70, 0x75, 0x76, 0x7f,
0x7c, 0x79, 0x7a, 0x3b, 0x38, 0x3d, 0x3e, 0x37, 0x34, 0x31, 0x32, 0x23, 0x20, 0x25, 0x26,
0x2f, 0x2c, 0x29, 0x2a, 0x0b, 0x08, 0x0d, 0x0e, 0x07, 0x04, 0x01, 0x02, 0x13, 0x10, 0x15,
0x16, 0x1f, 0x1c, 0x19, 0x1a}
```

3.5.3.7 table09

```
const byte table09[256] = {0x00, 0x09, 0x12, 0x1b, 0x24, 0x2d, 0x36, 0x3f, 0x48, 0x41, 0x5a,
0x53, 0x6c, 0x65, 0x7e, 0x77, 0x90, 0x99, 0x82, 0x8b, 0xb4, 0xbd, 0xa6, 0xaf, 0xd8, 0xd1,
0xca, 0xc3, 0xfc, 0xf5, 0xee, 0xe7, 0x3b, 0x32, 0x29, 0x20, 0x1f, 0x16, 0x0d, 0x04, 0x73,
0x7a, 0x61, 0x68, 0x57, 0x5e, 0x45, 0x4c, 0xab, 0xa2, 0xb9, 0xb0, 0x8f, 0x86, 0x9d, 0x94,
0xe3, 0xea, 0xf1, 0xf8, 0xc7, 0xce, 0xd5, 0xdc, 0x76, 0x7f, 0x64, 0x6d, 0x52, 0x5b, 0x40,
0x49, 0x3e, 0x37, 0x2c, 0x25, 0x1a, 0x13, 0x08, 0x01, 0xe6, 0xef, 0xf4, 0xfd, 0xc2, 0xcb,
0xd0, 0xd9, 0xae, 0xa7, 0xbc, 0xb5, 0x8a, 0x83, 0x98, 0x91, 0x4d, 0x44, 0x5f, 0x56, 0x69,
0x60, 0x7b, 0x72, 0x05, 0x0c, 0x17, 0x1e, 0x21, 0x28, 0x33, 0x3a, 0xdd, 0xd4, 0xcf, 0xc6,
0xf9, 0xf0, 0xeb, 0xe2, 0x95, 0x9c, 0x87, 0x8e, 0xb1, 0xb8, 0xa3, 0xaa, 0xec, 0xe5, 0xfe,
0xf7, 0xc8, 0xc1, 0xda, 0xd3, 0xa4, 0xad, 0xb6, 0xbf, 0x80, 0x89, 0x92, 0x9b, 0x7c, 0x75,
0x6e, 0x67, 0x58, 0x51, 0x4a, 0x43, 0x34, 0x3d, 0x26, 0x2f, 0x10, 0x19, 0x02, 0x0b, 0xd7,
0xde, 0xc5, 0xcc, 0xf3, 0xfa, 0xe1, 0xe8, 0x9f, 0x96, 0x8d, 0x84, 0xbb, 0xb2, 0xa9, 0xa0,
0x47, 0x4e, 0x55, 0x5c, 0x63, 0x6a, 0x71, 0x78, 0x0f, 0x06, 0x1d, 0x14, 0x2b, 0x22, 0x39,
0x30, 0x9a, 0x93, 0x88, 0x81, 0xbe, 0xb7, 0xac, 0xa5, 0xd2, 0xdb, 0xc0, 0xc9, 0xf6, 0xff,
0xe4, 0xed, 0x0a, 0x03, 0x18, 0x11, 0x2e, 0x27, 0x3c, 0x35, 0x42, 0x4b, 0x50, 0x59, 0x66,
0x6f, 0x74, 0x7d, 0xa1, 0xa8, 0xb3, 0xba, 0x85, 0x8c, 0x97, 0x9e, 0xe9, 0xe0, 0xfb, 0xf2,
0xcd, 0xc4, 0xdf, 0xd6, 0x31, 0x38, 0x23, 0x2a, 0x15, 0x1c, 0x07, 0x0e, 0x79, 0x70, 0x6b,
0x62, 0x5d, 0x54, 0x4f, 0x46}
```

3.5.3.8 table0b

```
const byte table0b[256] = {0x00, 0x0b, 0x16, 0x1d, 0x2c, 0x27, 0x3a, 0x31, 0x58, 0x53, 0x4e,
0x45, 0x74, 0x7f, 0x62, 0x69, 0xb0, 0xbb, 0xa6, 0xad, 0x9c, 0x97, 0x8a, 0x81, 0xe8, 0xe3,
0xfe, 0xf5, 0xc4, 0xcf, 0xd2, 0xd9, 0x7b, 0x70, 0x6d, 0x66, 0x57, 0x5c, 0x41, 0x4a, 0x23,
0x28, 0x35, 0x3e, 0x0f, 0x04, 0x19, 0x12, 0xcb, 0xc0, 0xdd, 0xd6, 0xe7, 0xec, 0xf1, 0xfa,
0x93, 0x98, 0x85, 0x8e, 0xbf, 0xb4, 0xa9, 0xa2, 0xf6, 0xfd, 0xe0, 0xeb, 0xda, 0xd1, 0xcc,
0xc7, 0xae, 0xa5, 0xb8, 0xb3, 0x82, 0x89, 0x94, 0x9f, 0x46, 0x4d, 0x50, 0x5b, 0x6a, 0x61,
0x7c, 0x77, 0x1e, 0x15, 0x08, 0x03, 0x32, 0x39, 0x24, 0x2f, 0x8d, 0x86, 0x9b, 0x90, 0xa1,
0xaa, 0xb7, 0xbc, 0xd5, 0xde, 0xc3, 0xc8, 0xf9, 0xf2, 0xef, 0xe4, 0x3d, 0x36, 0x2b, 0x20,
0x11, 0x1a, 0x07, 0x0c, 0x65, 0x6e, 0x73, 0x78, 0x49, 0x42, 0x5f, 0x54, 0xf7, 0xfc, 0xe1,
0xea, 0xdb, 0xd0, 0xcd, 0xc6, 0xaf, 0xa4, 0xb9, 0xb2, 0x83, 0x88, 0x95, 0x9e, 0x47, 0x4c,
0x51, 0x5a, 0x6b, 0x60, 0x7d, 0x76, 0x1f, 0x14, 0x09, 0x02, 0x33, 0x38, 0x25, 0x2e, 0x8c,
0x87, 0x9a, 0x91, 0xa0, 0xab, 0xb6, 0xbd, 0xd4, 0xdf, 0xc2, 0xc9, 0xf8, 0xf3, 0xee, 0xe5,
0x3c, 0x37, 0x2a, 0x21, 0x10, 0x1b, 0x06, 0x0d, 0x64, 0x6f, 0x72, 0x79, 0x48, 0x43, 0x5e,
0x55, 0x01, 0x0a, 0x17, 0x1c, 0x2d, 0x26, 0x3b, 0x30, 0x59, 0x52, 0x4f, 0x44, 0x75, 0x7e,
0x63, 0x68, 0xb1, 0xba, 0xa7, 0xac, 0x9d, 0x96, 0x8b, 0x80, 0xe9, 0xe2, 0xff, 0xf4, 0xc5,
0xce, 0xd3, 0xd8, 0x7a, 0x71, 0x6c, 0x67, 0x56, 0x5d, 0x40, 0x4b, 0x22, 0x29, 0x34, 0x3f,
0x0e, 0x05, 0x18, 0x13, 0xca, 0xc1, 0xdc, 0xd7, 0xe6, 0xed, 0xf0, 0xfb, 0x92, 0x99, 0x84,
0x8f, 0xbe, 0xb5, 0xa8, 0xa3}
```

3.5.3.9 table0d

```
const byte table0d[256] = {0x00, 0x0d, 0x1a, 0x17, 0x34, 0x39, 0x2e, 0x23, 0x68, 0x65, 0x72,
0x7f, 0x5c, 0x51, 0x46, 0x4b, 0xd0, 0xdd, 0xca, 0xc7, 0xe4, 0xe9, 0xfe, 0xf3, 0xb8, 0xb5,
0xa2, 0xaf, 0x8c, 0x81, 0x96, 0x9b, 0xbb, 0xb6, 0xa1, 0xac, 0x8f, 0x82, 0x95, 0x98, 0xd3,
0xde, 0xc9, 0xc4, 0xe7, 0xea, 0xfd, 0xf0, 0x6b, 0x66, 0x71, 0x7c, 0x5f, 0x52, 0x45, 0x48,
0x03, 0x0e, 0x19, 0x14, 0x37, 0x3a, 0x2d, 0x20, 0x6d, 0x60, 0x77, 0x7a, 0x59, 0x54, 0x43,
0x4e, 0x05, 0x08, 0x1f, 0x12, 0x31, 0x3c, 0x2b, 0x26, 0xbd, 0xb0, 0xa7, 0xaa, 0x89, 0x84,
0x93, 0x9e, 0xd5, 0xd8, 0xcf, 0xc2, 0xe1, 0xec, 0xfb, 0xf6, 0xd6, 0xdb, 0xcc, 0xc1, 0xe2,
0xef, 0xf8, 0xf5, 0xbe, 0xb3, 0xa4, 0xa9, 0x8a, 0x87, 0x90, 0x9d, 0x06, 0x0b, 0x1c, 0x11,
0x32, 0x3f, 0x28, 0x25, 0x6e, 0x63, 0x74, 0x79, 0x5a, 0x57, 0x40, 0x4d, 0xda, 0xd7, 0xc0,
0xcd, 0xee, 0xe3, 0xf4, 0xf9, 0xb2, 0xbf, 0xa8, 0xa5, 0x86, 0x8b, 0x9c, 0x91, 0x0a, 0x07,
0x10, 0x1d, 0x3e, 0x33, 0x24, 0x29, 0x62, 0x6f, 0x78, 0x75, 0x56, 0x5b, 0x4c, 0x41, 0x61,
0x6c, 0x7b, 0x76, 0x55, 0x58, 0x4f, 0x42, 0x09, 0x04, 0x13, 0x1e, 0x3d, 0x30, 0x27, 0x2a,
0xb1, 0xbc, 0xab, 0xa6, 0x85, 0x88, 0x9f, 0x92, 0xd9, 0xd4, 0xc3, 0xce, 0xed, 0xe0, 0xf7,
0xfa, 0xb7, 0xba, 0xad, 0xa0, 0x83, 0x8e, 0x99, 0x94, 0xdf, 0xd2, 0xc5, 0xc8, 0xeb, 0xe6,
0xf1, 0xfc, 0x67, 0x6a, 0x7d, 0x70, 0x53, 0x5e, 0x49, 0x44, 0x0f, 0x02, 0x15, 0x18, 0x3b,
0x36, 0x21, 0x2c, 0x0c, 0x01, 0x16, 0x1b, 0x38, 0x35, 0x22, 0x2f, 0x64, 0x69, 0x7e, 0x73,
0x50, 0x5d, 0x4a, 0x47, 0xdc, 0xd1, 0xc6, 0xcb, 0xe8, 0xe5, 0xf2, 0xff, 0xb4, 0xb9, 0xae,
0xa3, 0x80, 0x8d, 0x9a, 0x97}
```

3.5.3.10 table0e

```
const byte table0e[256] = {0x00, 0x0e, 0x1c, 0x12, 0x38, 0x36, 0x24, 0x2a, 0x70, 0x7e, 0x6c,
0x62, 0x48, 0x46, 0x54, 0x5a, 0xe0, 0xee, 0xfc, 0xf2, 0xd8, 0xd6, 0xc4, 0xca, 0x90, 0x9e,
0x8c, 0x82, 0xa8, 0xa6, 0xb4, 0xba, 0xdb, 0xd5, 0xc7, 0xc9, 0xe3, 0xed, 0xff, 0xf1, 0xab,
0xa5, 0xb7, 0xb9, 0x93, 0x9d, 0x8f, 0x81, 0x3b, 0x35, 0x27, 0x29, 0x03, 0x0d, 0x1f, 0x11,
0x4b, 0x45, 0x57, 0x59, 0x73, 0x7d, 0x6f, 0x61, 0xad, 0xa3, 0xb1, 0xbf, 0x95, 0x9b, 0x89,
0x87, 0xdd, 0xd3, 0xc1, 0xcf, 0xe5, 0xeb, 0xf9, 0xf7, 0x4d, 0x43, 0x51, 0x5f, 0x75, 0x7b,
0x69, 0x67, 0x3d, 0x33, 0x21, 0x2f, 0x05, 0x0b, 0x19, 0x17, 0x76, 0x78, 0x6a, 0x64, 0x4e,
0x40, 0x52, 0x5c, 0x06, 0x08, 0x1a, 0x14, 0x3e, 0x30, 0x22, 0x2c, 0x96, 0x98, 0x8a, 0x84,
0xae, 0xa0, 0xb2, 0xbc, 0xe6, 0xe8, 0xfa, 0xf4, 0xde, 0xd0, 0xc2, 0xcc, 0x41, 0x4f, 0x5d,
0x53, 0x79, 0x77, 0x65, 0x6b, 0x31, 0x3f, 0x2d, 0x23, 0x09, 0x07, 0x15, 0x1b, 0xa1, 0xaf,
0xbd, 0xb3, 0x99, 0x97, 0x85, 0x8b, 0xd1, 0xdf, 0xcd, 0xc3, 0xe9, 0xe7, 0xf5, 0xfb, 0x9a,
0x94, 0x86, 0x88, 0xa2, 0xac, 0xbe, 0xb0, 0xea, 0xe4, 0xf6, 0xf8, 0xd2, 0xdc, 0xce, 0xc0,
0x7a, 0x74, 0x66, 0x68, 0x42, 0x4c, 0x5e, 0x50, 0x0a, 0x04, 0x16, 0x18, 0x32, 0x3c, 0x2e,
0x20, 0xec, 0xe2, 0xf0, 0xfe, 0xd4, 0xda, 0xc8, 0xc6, 0x9c, 0x92, 0x80, 0x8e, 0xa4, 0xaa,
0xb8, 0xb6, 0x0c, 0x02, 0x10, 0x1e, 0x34, 0x3a, 0x28, 0x26, 0x7c, 0x72, 0x60, 0x6e, 0x44,
0x4a, 0x58, 0x56, 0x37, 0x39, 0x2b, 0x25, 0x0f, 0x01, 0x13, 0x1d, 0x47, 0x49, 0x5b, 0x55,
0x7f, 0x71, 0x63, 0x6d, 0xd7, 0xd9, 0xcb, 0xc5, 0xef, 0xe1, 0xf3, 0xfd, 0xa7, 0xa9, 0xbb,
0xb5, 0x9f, 0x91, 0x83, 0x8d}
```

3.6 Référence du fichier c/cipher.h

Function prototypes of the cipher method.

Définitions de type

- typedef unsigned char byte

Fonctions

- void addRoundKey (byte state[ ], byte w[ ], int round)
  Add the key to the state (xor operation)
- void subBytes (byte state[ ], const byte box[256], int length)
  Substitute the bytes of the state with a box.
- void shiftOneRow (byte state[ ], int row, int direction, int shift)
  Shift one row of the state.
- void shiftRows (byte state[ ])
  Shift all the rows of the state.
- void invShiftRows (byte state[ ])
  Inverse process of shiftRows.
- void mixColumns (byte state[ ], const int inv)
  Mix the columns of the state.
- void rotWord (byte state[4])
  1 byte rigth rotation of a 4 byte state
- void rcon (int i, byte out[4])
  Create the rcon polynome associated to the round.
- void keyExpansion (byte key[ ], byte w[ ], int nk, int nr)
  Key expansion method.
- void cipher (byte in[ ], byte w[ ], int nr)
  Cipher method.
- void invCipher (byte in[ ], byte w[ ], int nr)
  Inverse cipher method.

3.6.1 Description détaillée

Function prototypes of the cipher method.

Contient les prototypes pour le cipher

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.6.2 Documentation des définitions de type

3.6.2.1 byte

```
typedef unsigned char byte
```

3.6.3 Documentation des fonctions

3.6.3.1 addRoundKey()

```
void addRoundKey (
             byte state[ ],
             byte w[ ],
             int round )
```

Add the key to the state (xor operation)

Paramètres

state	The current state (16 bytes)
w	The entire key
round	The current round (relative to Nr)

Renvoie

Void

3.6.3.2 cipher()

```
void cipher (
            byte in[ ],
            byte w[ ],
            int nr )
```

Cipher method.

Paramètres

in	The input block (16 bytes) enlarged over the rounds
w	The expanded key (16∗(Nr+1) bytes)
nr	The number of rounds

Renvoie

Void

3.6.3.3 invCipher()

```
void invCipher (
            byte in[ ],
            byte w[ ],
            int nr )
```

Inverse cipher method.

Paramètres

in	The input block (16 bytes) enlarged over the rounds
w	The expanded key (16∗(Nr+1) bytes)
nr	The number of rounds

Renvoie

Void

3.6.3.4 invShiftRows()

```
void invShiftRows (
            byte state[] )
```

Inverse process of shiftRows.

Paramètres

state	The current state (16 bytes)

Renvoie

Void

3.6.3.5 keyExpansion()

```
void keyExpansion (
            byte key[],
            byte w[],
            int nk,
            int nr )
```

Key expansion method.

Paramètres

key	The key (16, 24 or 32 bytes)
w	The expanded key generated (16∗(Nr+1) bytes)
nk	The number of words in the key (4, 6 or 8 refering to the key size (16, 24 or 32 bytes)))
nr	The number of rounds

Renvoie

Void

3.6.3.6 mixColumns()

```
void mixColumns (
            byte state[],
            const int inv )
```

Mix the columns of the state.

Paramètres

state	The current state (16 bytes)
inv	1 for the Inverse Mix Columns, 0 for the Mix Columns

Renvoie

Void

3.6.3.7 rcon()

```
void rcon (
            int i,
            byte out[4] )
```

Create the rcon polynome associated to the round.

Paramètres

i	The current round
out	The word generated (4 bytes)

Renvoie

Void

3.6.3.8 rotWord()

```
void rotWord (
            byte state[4] )
```

1 byte rigth rotation of a 4 byte state

Paramètres

state	The current word (4 bytes)

Renvoie

Void

3.6.3.9 shiftOneRow()

```
void shiftOneRow (
            byte state[],
            int row,
            int direction,
            int shift )
```

Shift one row of the state.

Paramètres

state	The current state (16 bytes)
row	The row to shift
direction	The direction of the shift (1 for right, -1 for left)
shift	The number of shifts

Renvoie

Void

3.6.3.10 shiftRows()

```
void shiftRows (
            byte state[] )
```

Shift all the rows of the state.

Paramètres

state	The current state (16 bytes)

Renvoie

Void

3.6.3.11 subBytes()

```
void subBytes (
            byte state[],
            const byte box[256],
            int length )
```

Substitute the bytes of the state with a box.

Paramètres

state	The current state (16 bytes)
box	Either the S-Box or the inverse S-Box (256 bytes)
length	The length of the state (16 for subBytes and 4 for subWord)

Renvoie

Void

3.7 cipher.h

Aller à la documentation de ce fichier.

```

/* -- Defines -- */
typedef unsigned char byte;


/* -- Functions -- */
void addRoundKey(byte state[], byte w[], int round);

void subBytes(byte state[], const byte box[256], int length);

void shiftOneRow(byte state[], int row, int direction, int shift);

void shiftRows(byte state[]);

void invShiftRows(byte state[]);

void mixColumns(byte state[], const int inv);

void rotWord(byte state[4]);

void rcon(int i, byte out[4]);

void keyExpansion(byte key[], byte w[], int nk, int nr);

void cipher(byte in[], byte w[], int nr);

void invCipher(byte in[], byte w[], int nr);
```

3.8 Référence du fichier c/entropie.c

Entropy algorithm.

```
#include <stdio.h>
#include <stdlib.h>
#include <math.h>
```

Définitions de type

- typedef unsigned char byte

Fonctions

- void entropie (char *filename)
  Calculate the entropy of a file.

3.8.1 Description détaillée

Entropy algorithm.

Calcule à quel point c'est le ¨chaos¨ dans le fichier. Observation de la répartition des octets : s'ils sont tout présent uniformément, l'entropie est maximale.

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.8.2 Documentation des définitions de type

3.8.2.1 byte

```
typedef unsigned char byte
```

3.8.3 Documentation des fonctions

3.8.3.1 entropie()

```
void entropie (
            char * filename )
```

Calculate the entropy of a file.

Paramètres

filename	The filename of the file

Renvoie

Void

3.9 Référence du fichier c/entropie.h

Function prototypes of the entropy algorithm.

Fonctions

- void entropie (char ∗filename)
  Calculate the entropy of a file.

3.9.1 Description détaillée

Function prototypes of the entropy algorithm.

Contient les prototypes pour l'algorithme d'entropie

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.9.2 Documentation des fonctions

3.9.2.1 entropie()

```
void entropie (
             char ∗ filename )
```

Calculate the entropy of a file.

Paramètres

filename	The filename of the file

Renvoie

Void

3.10 entropie.h

Aller à la documentation de ce fichier.

```

/* -- Functions -- */
void entropie(char *filename);
```

3.11 Référence du fichier c/tests.c

Tests methods.

```
#include ¨cipher.h¨
#include ¨tools.h¨
#include ¨aes.h¨
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
```

Fonctions

- void testByteXor (void)
- void testMulti (void)
- void testSwitchColRows (void)
- void testSplitArr (void)
- void testMergeArr (void)
- void testAddRoundKey (void)
- void testSubBytes (void)
- void testShiftRows (void)
- void testInvShiftRows (void)
- void testMixColums (void)
- void testInvMixColums (void)
- void testSubWord (void)
- void testRotWord (void)
- void testRcon (void)
- void testKeyExpansion (void)
- void testCipher (void)
- void testInvCipher (void)
- void testHexToAscii (void)
- void testAsciiToHex (void)
- void testAesEncrypt (void)
- long getFileSize (FILE ∗file)
- char ∗ readFromFile (const char ∗filename)
- void writeFile (const char ∗filename, const char ∗content)
- void testAesEncryptFile (void)
- void testAesDecrypt (void)
- void testAesDecryptFile (void)
- int main (void)

  Main function.

Variables

- const byte sbox_test [256] = {0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67, 0x2b, 0xfe, 0xd7, 0xab, 0x76, 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4, 0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, 0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34, 0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15, 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a, 0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, 0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a, 0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84, 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc, 0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, 0xd0, 0xef, 0xaa, 0xfb, 0x43, 0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0x51, 0xa3, 0x40, 0x8f, 0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, 0xcd, 0x0c, 0x13, 0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73, 0x60, 0x81, 0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, 0xe0, 0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79, 0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae, 0x08, 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd, 0x8b, 0x8a, 0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86, 0xc1, 0x1d, 0x9e, 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9, 0xce, 0x55, 0x28, 0xdf, 0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d, 0x0f, 0xb0, 0x54, 0xbb, 0x16}
- const byte invSbox_test [256] = {0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40, 0xa3, 0x9e, 0x81, 0xf3, 0xd7, 0xfb, 0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87, 0x34, 0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb, 0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d, 0xee, 0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e, 0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24, 0xb2, 0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25, 0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68, 0x98, 0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92, 0x6c, 0x70, 0x48, 0x50, 0xfd, 0xed, 0xb9, 0xda, 0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84, 0x90, 0xd8, 0xab, 0x00, 0x8c, 0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06, 0xd0, 0x2c, 0x1e, 0x8f, 0xca, 0x3f, 0x0f, 0x02, 0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b, 0x3a, 0x91, 0x11, 0x41, 0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73, 0x96, 0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e, 0x47, 0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe, 0x1b, 0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd, 0x5a, 0xf4, 0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27, 0x80, 0xec, 0x5f, 0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f, 0x93, 0xc9, 0x9c, 0xef, 0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0, 0xc8, 0xeb, 0xbb, 0x3c, 0x83, 0x53, 0x99, 0x61, 0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69, 0x14, 0x63, 0x55, 0x21, 0x0c, 0x7d}

3.11.1 Description détaillée

Tests methods.

Réalise un ensemble de tests unitaires sur les fonctions de notre code afin de s'assurer de leur bon fonctionnement.

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.11.2 Documentation des fonctions

3.11.2.1 getFileSize()

```
long getFileSize (
            FILE * file )
```

3.11.2.2 main()

```
int main (
            void )
```

Main function.

Appel de toutes les fonctions de test

Renvoie

Void

3.11.2.3 readFromFile()

```
char * readFromFile (
            const char * filename )
```

3.11.2.4 testAddRoundKey()

```
void testAddRoundKey (
            void )
```

3.11.2.5 testAesDecrypt()

```
void testAesDecrypt (
            void )
```

3.11.2.6 testAesDecryptFile()

```
void testAesDecryptFile (
            void )
```

3.11.2.7 testAesEncrypt()

```
void testAesEncrypt (
            void )
```

3.11.2.8 testAesEncryptFile()

```
void testAesEncryptFile (
            void )
```

3.11.2.9 testAsciiToHex()

```
void testAsciiToHex (
            void  )
```

3.11.2.10 testByteXor()

```
void testByteXor (
            void  )
```

3.11.2.11 testCipher()

```
void testCipher (
            void  )
```

3.11.2.12 testHexToAscii()

```
void testHexToAscii (
            void  )
```

3.11.2.13 testInvCipher()

```
void testInvCipher (
            void  )
```

3.11.2.14 testInvMixColums()

```
void testInvMixColums (
            void  )
```

3.11.2.15 testInvShiftRows()

```
void testInvShiftRows (
            void  )
```

3.11.2.16 testKeyExpansion()

```
void testKeyExpansion (
            void  )
```

3.11.2.17 testMergeArr()

```
void testMergeArr (
            void  )
```

3.11.2.18 testMixColums()

```
void testMixColums (
             void  )
```

3.11.2.19 testMulti()

```
void testMulti (
             void  )
```

3.11.2.20 testRcon()

```
void testRcon (
             void  )
```

3.11.2.21 testRotWord()

```
void testRotWord (
             void  )
```

3.11.2.22 testShiftRows()

```
void testShiftRows (
             void  )
```

3.11.2.23 testSplitArr()

```
void testSplitArr (
             void  )
```

3.11.2.24 testSubBytes()

```
void testSubBytes (
             void  )
```

3.11.2.25 testSubWord()

```
void testSubWord (
             void  )
```

3.11.2.26 testSwitchColRows()

```
void testSwitchColRows (
             void  )
```

3.11.2.27 writeFile()

```
void writeFile (
            const char * filename,
            const char * content )
```

3.11.3 Documentation des variables

3.11.3.1 invSbox_test

```
const byte invSbox_test[256] = {0x52, 0x09, 0x6a, 0xd5, 0x30, 0x36, 0xa5, 0x38, 0xbf, 0x40,
0xa3, 0x9e, 0x81, 0xf3, 0xd7, 0xfb, 0x7c, 0xe3, 0x39, 0x82, 0x9b, 0x2f, 0xff, 0x87, 0x34,
0x8e, 0x43, 0x44, 0xc4, 0xde, 0xe9, 0xcb, 0x54, 0x7b, 0x94, 0x32, 0xa6, 0xc2, 0x23, 0x3d,
0xee, 0x4c, 0x95, 0x0b, 0x42, 0xfa, 0xc3, 0x4e, 0x08, 0x2e, 0xa1, 0x66, 0x28, 0xd9, 0x24,
0xb2, 0x76, 0x5b, 0xa2, 0x49, 0x6d, 0x8b, 0xd1, 0x25, 0x72, 0xf8, 0xf6, 0x64, 0x86, 0x68,
0x98, 0x16, 0xd4, 0xa4, 0x5c, 0xcc, 0x5d, 0x65, 0xb6, 0x92, 0x6c, 0x70, 0x48, 0x50, 0xfd,
0xed, 0xb9, 0xda, 0x5e, 0x15, 0x46, 0x57, 0xa7, 0x8d, 0x9d, 0x84, 0x90, 0xd8, 0xab, 0x00,
0x8c, 0xbc, 0xd3, 0x0a, 0xf7, 0xe4, 0x58, 0x05, 0xb8, 0xb3, 0x45, 0x06, 0xd0, 0x2c, 0x1e,
0x8f, 0xca, 0x3f, 0x0f, 0x02, 0xc1, 0xaf, 0xbd, 0x03, 0x01, 0x13, 0x8a, 0x6b, 0x3a, 0x91,
0x11, 0x41, 0x4f, 0x67, 0xdc, 0xea, 0x97, 0xf2, 0xcf, 0xce, 0xf0, 0xb4, 0xe6, 0x73, 0x96,
0xac, 0x74, 0x22, 0xe7, 0xad, 0x35, 0x85, 0xe2, 0xf9, 0x37, 0xe8, 0x1c, 0x75, 0xdf, 0x6e,
0x47, 0xf1, 0x1a, 0x71, 0x1d, 0x29, 0xc5, 0x89, 0x6f, 0xb7, 0x62, 0x0e, 0xaa, 0x18, 0xbe,
0x1b, 0xfc, 0x56, 0x3e, 0x4b, 0xc6, 0xd2, 0x79, 0x20, 0x9a, 0xdb, 0xc0, 0xfe, 0x78, 0xcd,
0x5a, 0xf4, 0x1f, 0xdd, 0xa8, 0x33, 0x88, 0x07, 0xc7, 0x31, 0xb1, 0x12, 0x10, 0x59, 0x27,
0x80, 0xec, 0x5f, 0x60, 0x51, 0x7f, 0xa9, 0x19, 0xb5, 0x4a, 0x0d, 0x2d, 0xe5, 0x7a, 0x9f,
0x93, 0xc9, 0x9c, 0xef, 0xa0, 0xe0, 0x3b, 0x4d, 0xae, 0x2a, 0xf5, 0xb0, 0xc8, 0xeb, 0xbb,
0x3c, 0x83, 0x53, 0x99, 0x61, 0x17, 0x2b, 0x04, 0x7e, 0xba, 0x77, 0xd6, 0x26, 0xe1, 0x69,
0x14, 0x63, 0x55, 0x21, 0x0c, 0x7d}
```

3.11.3.2 sbox_test

```
const byte sbox_test[256] = {0x63, 0x7c, 0x77, 0x7b, 0xf2, 0x6b, 0x6f, 0xc5, 0x30, 0x01, 0x67,
0x2b, 0xfe, 0xd7, 0xab, 0x76, 0xca, 0x82, 0xc9, 0x7d, 0xfa, 0x59, 0x47, 0xf0, 0xad, 0xd4,
0xa2, 0xaf, 0x9c, 0xa4, 0x72, 0xc0, 0xb7, 0xfd, 0x93, 0x26, 0x36, 0x3f, 0xf7, 0xcc, 0x34,
0xa5, 0xe5, 0xf1, 0x71, 0xd8, 0x31, 0x15, 0x04, 0xc7, 0x23, 0xc3, 0x18, 0x96, 0x05, 0x9a,
0x07, 0x12, 0x80, 0xe2, 0xeb, 0x27, 0xb2, 0x75, 0x09, 0x83, 0x2c, 0x1a, 0x1b, 0x6e, 0x5a,
0xa0, 0x52, 0x3b, 0xd6, 0xb3, 0x29, 0xe3, 0x2f, 0x84, 0x53, 0xd1, 0x00, 0xed, 0x20, 0xfc,
0xb1, 0x5b, 0x6a, 0xcb, 0xbe, 0x39, 0x4a, 0x4c, 0x58, 0xcf, 0xd0, 0xef, 0xaa, 0xfb, 0x43,
0x4d, 0x33, 0x85, 0x45, 0xf9, 0x02, 0x7f, 0x50, 0x3c, 0x9f, 0xa8, 0x51, 0xa3, 0x40, 0x8f,
0x92, 0x9d, 0x38, 0xf5, 0xbc, 0xb6, 0xda, 0x21, 0x10, 0xff, 0xf3, 0xd2, 0xcd, 0x0c, 0x13,
0xec, 0x5f, 0x97, 0x44, 0x17, 0xc4, 0xa7, 0x7e, 0x3d, 0x64, 0x5d, 0x19, 0x73, 0x60, 0x81,
0x4f, 0xdc, 0x22, 0x2a, 0x90, 0x88, 0x46, 0xee, 0xb8, 0x14, 0xde, 0x5e, 0x0b, 0xdb, 0xe0,
0x32, 0x3a, 0x0a, 0x49, 0x06, 0x24, 0x5c, 0xc2, 0xd3, 0xac, 0x62, 0x91, 0x95, 0xe4, 0x79,
0xe7, 0xc8, 0x37, 0x6d, 0x8d, 0xd5, 0x4e, 0xa9, 0x6c, 0x56, 0xf4, 0xea, 0x65, 0x7a, 0xae,
0x08, 0xba, 0x78, 0x25, 0x2e, 0x1c, 0xa6, 0xb4, 0xc6, 0xe8, 0xdd, 0x74, 0x1f, 0x4b, 0xbd,
0x8b, 0x8a, 0x70, 0x3e, 0xb5, 0x66, 0x48, 0x03, 0xf6, 0x0e, 0x61, 0x35, 0x57, 0xb9, 0x86,
0xc1, 0x1d, 0x9e, 0xe1, 0xf8, 0x98, 0x11, 0x69, 0xd9, 0x8e, 0x94, 0x9b, 0x1e, 0x87, 0xe9,
0xce, 0x55, 0x28, 0xdf, 0x8c, 0xa1, 0x89, 0x0d, 0xbf, 0xe6, 0x42, 0x68, 0x41, 0x99, 0x2d,
0x0f, 0xb0, 0x54, 0xbb, 0x16}
```

3.12 Référence du fichier c/tools.c

Tools method.

```
#include <string.h>
#include <stdio.h>
```

Définitions de type

- typedef unsigned char byte

Fonctions

- void byteXor (byte a[ ], const byte b[ ], int length)
  XOR operation between two byte arrays.
- byte multi (byte a, byte b)
  Multiplication in GF(2^8) for two bytes.
- void printByte (byte in[ ], int length)
  Print a byte array.
- void switchColRows (byte state[ ])
  Switch the columns and the rows of a 4x4 matrix.
- void splitArr (const byte in[ ], byte out[ ], int start, int end)
  Split an array into another one.
- void mergeArr (const byte in[ ], byte out[ ], int start, int end)
  Merge an array into another one (use for append an array)

3.12.1 Description détaillée

Tools method.

Contient un emsemble de fonctions utiles pour tout le reste de notre code Ex: affichage, séparation de tableau...

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.12.2 Documentation des définitions de type

3.12.2.1 byte

```
typedef unsigned char byte
```

3.12.3 Documentation des fonctions

3.12.3.1 byteXor()

```
void byteXor (
            byte a[ ],
            const byte b[ ],
            int length )
```

XOR operation between two byte arrays.

Paramètres

a	First byte array
b	Second byte array
length	Length of the arrays

Renvoie

Void

3.12.3.2 mergeArr()

```
void mergeArr (
             const byte in[],
             byte out[],
             int start,
             int end )
```

Merge an array into another one (use for append an array)

Paramètres

in	The array to merge
out	The array to fill
start	The start index (for the ¨out¨ array)
end	The end index (for the ¨out¨ array)

Renvoie

Void

3.12.3.3 multi()

```
byte multi (
             byte a,
             byte b )
```

Multiplication in GF(2^8) for two bytes.

Paramètres

a	First byte
b	Second byte

Renvoie

The result of the multiplication

3.12.3.4 printByte()

```
void printByte (
             byte in[],
             int length )
```

Print a byte array.

Paramètres

in	The byte array to print
length	The length of the array

Renvoie

Void

3.12.3.5 splitArr()

```
void splitArr (
             const byte in[],
             byte out[],
             int start,
             int end )
```

Split an array into another one.

Paramètres

in	The array to split
out	The array to fill
start	The start index (for the ¨in¨ array)
end	The end index (for the ¨in¨ array)

Renvoie

Void

3.12.3.6 switchColRows()

```
void switchColRows (
             byte state[] )
```

Switch the columns and the rows of a 4x4 matrix.

Paramètres

state	The matrix to switch

Renvoie

Void

3.13 Référence du fichier c/tools.h

Functions prototypes of the tools.c file.

Définitions de type

- typedef unsigned char byte

Fonctions

- void byteXor (byte a[ ], const byte b[ ], int length)
  XOR operation between two byte arrays.
- byte multi (byte a, byte b)
  Multiplication in GF(2^8) for two bytes.
- void printByte (byte in[ ], int length)
  Print a byte array.
- void switchColRows (byte state[ ])
  Switch the columns and the rows of a 4x4 matrix.
- void splitArr (const byte in[ ], byte out[ ], int start, int end)
  Split an array into another one.
- void mergeArr (const byte in[ ], byte out[ ], int start, int end)
  Merge an array into another one (use for append an array)

3.13.1 Description détaillée

Functions prototypes of the tools.c file.

Contient les prototypes des fonctions de tools.c

Auteur

Mazzone Rémi (rems-38)
Moussu Guillemot (guillemotmoussu)

Bogue No known bugs.

3.13.2 Documentation des définitions de type

3.13.2.1 byte

```
typedef unsigned char byte
```

3.13.3 Documentation des fonctions

3.13.3.1 byteXor()

```
void byteXor (
             byte a[ ],
             const byte b[ ],
             int length )
```

XOR operation between two byte arrays.

Paramètres

a	First byte array
b	Second byte array
length	Length of the arrays

Renvoie

Void

3.13.3.2 mergeArr()

```
void mergeArr (
             const byte in[],
             byte out[],
             int start,
             int end )
```

Merge an array into another one (use for append an array)

Paramètres

in	The array to merge
out	The array to fill
start	The start index (for the ¨out¨ array)
end	The end index (for the ¨out¨ array)

Renvoie

Void

3.13.3.3 multi()

```
byte multi (
             byte a,
             byte b )
```

Multiplication in GF(2^8) for two bytes.

Paramètres

a	First byte
b	Second byte

Renvoie

The result of the multiplication

3.13.3.4 printByte()

```
void printByte (
             byte in[],
             int length )
```

Print a byte array.

Paramètres

in	The byte array to print
length	The length of the array

Renvoie

Void

3.13.3.5 splitArr()

```
void splitArr (
             const byte in[],
             byte out[],
             int start,
             int end )
```

Split an array into another one.

Paramètres

in	The array to split
out	The array to fill
start	The start index (for the ¨in¨ array)
end	The end index (for the ¨in¨ array)

Renvoie

Void

3.13.3.6 switchColRows()

```
void switchColRows (
             byte state[] )
```

Switch the columns and the rows of a 4x4 matrix.

Paramètres

state	The matrix to switch

Renvoie

Void

3.14 tools.h

Aller à la documentation de ce fichier.

```
/* -- Defines -- */
typedef unsigned char byte;


/* -- Functions -- */
void byteXor(byte a[], const byte b[], int length);

byte multi(byte a, byte b);

void printByte(byte in[], int length);

void switchColRows(byte state[]);

void splitArr(const byte in[], byte out[], int start, int end);

void mergeArr(const byte in[], byte out[], int start, int end);
```

Index