Universidad Nacional Autónoma de México

Facultad de Ingeniería

CRIPTOGRAFÍA

Exámen Parcial 2 Parte: Cifrado AES-128

Grupo 1

Integrantes:

Banda Martínez César Eduardo

Limón Hernández Raúl Rogelio

2018-1

Funciones requeridas:

SubBytes: Se utiliza nuevamente la caja S empleada en la expansión de llaves, pero esta vez se desempeñarán corrimientos a la derecha de 4 bits para obtener la primera coordenada y para la otra se filtrará mediante una máscara y una operación AND. Esta operación para todas las celdas de la matriz de estado.

```
289
290  void SubBytes(B8 in[4][4]){
291
292      B8 X_BOX[16][16]={{0x63,0x7c,0x77,0x7b,0xf2,0x6b,0x6f,0xc5,0x30,0x01,0x67,0x2b,0xfe,0xd7,0xab,0x76},
293                        {0xca,0x82,0xc9,0x7d,0xfa,0x59,0x47,0xf0,0xad,0xd4,0xa2,0xaf,0x9c,0xa4,0x72,0xc0},
294                        {0xb7,0xfd,0x93,0x26,0x36,0x3f,0xf7,0xcc,0x34,0xa5,0xe5,0xf1,0x71,0xd8,0x31,0x15},
295                        {0x04,0xc7,0x23,0xc3,0x18,0x96,0x05,0x9a,0x07,0x12,0x80,0xe2,0xeb,0x27,0xb2,0x75},
296                        {0x09,0x83,0x2c,0x1a,0x1b,0x6e,0x5a,0xa0,0x52,0x3b,0xd6,0xb3,0x29,0xe3,0x2f,0x84},
297                        {0x53,0xd1,0x00,0xed,0x20,0xfc,0xb1,0x5b,0x6a,0xcb,0xbe,0x39,0x4a,0x4c,0x58,0xcf},
298                        {0xd0,0xef,0xaa,0xfb,0x43,0x4d,0x33,0x85,0x45,0xf9,0x02,0x7f,0x50,0x3c,0x9f,0xa8},
299                        {0x51,0xa3,0x40,0x8f,0x92,0x9d,0x38,0xf5,0xbc,0xb6,0xda,0x21,0x10,0xff,0xf3,0xd2},
300                        {0xcd,0x0c,0x13,0xec,0x5f,0x97,0x44,0x17,0xc4,0xa7,0x7e,0x3d,0x64,0x5d,0x19,0x73},
301                        {0x60,0x81,0x4f,0xdc,0x22,0x2a,0x90,0x88,0x46,0xee,0xb8,0x14,0xde,0x5e,0x0b,0xdb},
302                        {0xe0,0x32,0x3a,0x0a,0x49,0x06,0x24,0x5c,0xc2,0xd3,0xac,0x62,0x91,0x95,0xe4,0x79},
303                        {0xe7,0xc8,0x37,0x6d,0x8d,0xd5,0x4e,0xa9,0x6c,0x56,0xf4,0xea,0x65,0x7a,0xae,0x08},
304                        {0xba,0x78,0x25,0x2e,0x1c,0xa6,0xb4,0xc6,0xe8,0xdd,0x74,0x1f,0x4b,0xbd,0x8b,0x8a},
305                        {0x70,0x3e,0xb5,0x66,0x48,0x03,0xf6,0x0e,0x61,0x35,0x57,0xb9,0x86,0xc1,0x1d,0x9e},
306                        {0xe1,0xf8,0x98,0x11,0x69,0xd9,0x8e,0x94,0x9b,0x1e,0x87,0xe9,0xce,0x55,0x28,0xdf},
307                        {0x8c,0xa1,0x89,0x0d,0xbf,0xe6,0x42,0x68,0x41,0x99,0x2d,0x0f,0xb0,0x54,0xbb,0x16}
308                         };
309
310      B32 arr;
311      int i;
312
313      int coord[8];
314
315      for(i=0;i<4;i++){
316          if(i==0){
317              in[i][0]=X_BOX[in[i][0]>>4][in[i][0] & (0x0f)];
318              in[i][1]=X_BOX[in[i][1]>>4][in[i][1] & (0x0f)];
319              in[i][2]=X_BOX[in[i][2]>>4][in[i][2] & (0x0f)];
```

```
313      int coord[8];
314
315      for(i=0;i<4;i++){
316          if(i==0){
317              in[i][0]=X_BOX[in[i][0]>>4][in[i][0] & (0x0f)];
318              in[i][1]=X_BOX[in[i][1]>>4][in[i][1] & (0x0f)];
319              in[i][2]=X_BOX[in[i][2]>>4][in[i][2] & (0x0f)];
320              in[i][3]=X_BOX[in[i][3]>>4][in[i][3] & (0x0f)];
321          }
322
323          if(i==1){
324              in[i][0]=X_BOX[in[i][0]>>4][in[i][0] & (0x0f)];
325              in[i][1]=X_BOX[in[i][1]>>4][in[i][1] & (0x0f)];
326              in[i][2]=X_BOX[in[i][2]>>4][in[i][2] & (0x0f)];
327              in[i][3]=X_BOX[in[i][3]>>4][in[i][3] & (0x0f)];
328          }
329
330          if(i==2){
331              in[i][0]=X_BOX[in[i][0]>>4][in[i][0] & (0x0f)];
332              in[i][1]=X_BOX[in[i][1]>>4][in[i][1] & (0x0f)];
333              in[i][2]=X_BOX[in[i][2]>>4][in[i][2] & (0x0f)];
334              in[i][3]=X_BOX[in[i][3]>>4][in[i][3] & (0x0f)];
335          }
336
337          if(i==3){
338              in[i][0]=X_BOX[in[i][0]>>4][in[i][0] & (0x0f)];
339              in[i][1]=X_BOX[in[i][1]>>4][in[i][1] & (0x0f)];
340              in[i][2]=X_BOX[in[i][2]>>4][in[i][2] & (0x0f)];
341              in[i][3]=X_BOX[in[i][3]>>4][in[i][3] & (0x0f)];
342          }
343
```

ShiftRows: Se acude a un vector de 4 elementos que almacene de forma temporal los elementos por renglón, para luego hacer las respectivas igualaciones para poder implementar el corrimiento de los renglones.

```
243  void ShiftRows(B8 in[4][4]){
244
245      int i,v;
246      B8 vector[4];
247
248      for(i=1;i<4;i++){
249
250          if(i==1){
251
252              for(v=0;v<4;v++)
253                  vector[v]=in[i][v];
254
255              in[i][0]=vector[1];
256              in[i][1]=vector[2];
257              in[i][2]=vector[3];
258              in[i][3]=vector[0];
259
260          }
261
262          if(i==2){
263
264              for(v=0;v<4;v++)
265                  vector[v]=in[i][v];
266
267              in[i][0]=vector[2];
268              in[i][1]=vector[3];
269              in[i][2]=vector[0];
270              in[i][3]=vector[1];
271
272          }
273
274          if(i==3){
275
276              for(v=0;v<4;v++)
277                  vector[v]=in[i][v];
278
279              in[i][0]=vector[3];
280              in[i][1]=vector[0];
281              in[i][2]=vector[1];
282              in[i][3]=vector[2];
283
284          }
285
286      }
```

MixColumns: Se copia la matriz de estado en una matriz bidimensional temporal. Se efectúa el producto matricial, columna por columna, basándose para ello en la función FF.

```
166  void MixColumns(B8 in[4][4]){
167
168    int i=0,j;
169
170    B8 coef2=0x02;
171    B8 coef3=0x03;
172
173    B16 temp[4][4];
174
175    for(i=0;i<4;i++)
176      for(j=0;j<4;j++)
177          temp[i][j]=in[i][j];
178
179    for(i=0;i<4;i++){
180      in[0][i]=FF(temp[0][i],coef2)^FF(temp[1][i],coef3)^temp[2][i]^temp[3][i];
181      in[1][i]=temp[0][i]^FF(temp[1][i],coef2)^FF(temp[2][i],coef3)^temp[3][i];
182      in[2][i]=temp[0][i]^temp[1][i]^FF(temp[2][i],coef2)^FF(temp[3][i],coef3);
183      in[3][i]=temp[1][i]^temp[2][i]^FF(temp[3][i],coef2)^FF(temp[0][i],coef3);
184    }
185
```

FF: Siglas de “Finite Field”, es la función encargada de desempeñar el producto matricial apoyándose en las tablas “L” y “E”, que además conservará el resultado del producto dentro del campo finito en módulo 0xff.

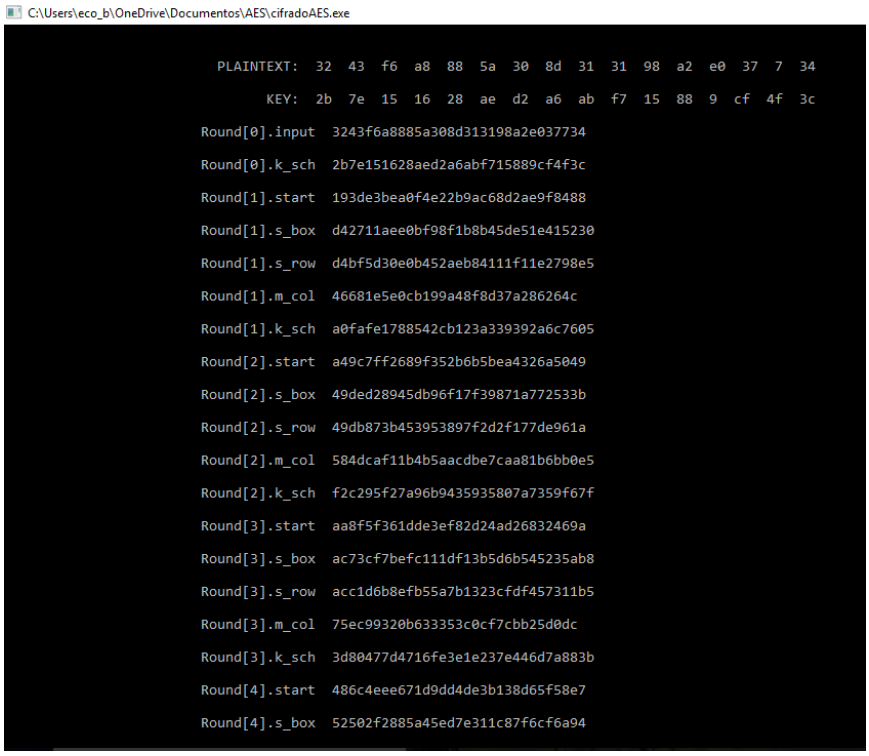
```
188  B16 FF(B16 celda, B8 coef){
189
190    B8 L[16][16]={{0x00,0x00,0x19,0x01,0x32,0x02,0x1a,0xc6,0x4b,0xc7,0x1b,0x68,0x33,0xee,0xdf,0x03},
191                  {0x64,0x04,0xe0,0x0e,0x34,0x8d,0x81,0xef,0x4c,0x71,0x08,0xc8,0xf8,0x69,0x1c,0xc1},
192                  {0x7d,0xc2,0x1d,0xb5,0xf9,0xb9,0x27,0x6a,0x4d,0xe4,0xa6,0x72,0x9a,0xc9,0x09,0x78},
193                  {0x65,0x2f,0x8a,0x05,0x21,0x0f,0xe1,0x24,0x12,0xf0,0x82,0x45,0x35,0x93,0xda,0x8e},
194                  {0x96,0x8f,0xdb,0xbd,0x36,0xd0,0xce,0x94,0x13,0x5c,0xd2,0xf1,0x40,0x46,0x83,0x38},
195                  {0x66,0xdd,0xfd,0x30,0xbf,0x06,0x8b,0x62,0xb3,0x25,0xe2,0x98,0x22,0x88,0x91,0x10},
196                  {0x7e,0x6e,0x48,0xc3,0xa3,0xb6,0x1e,0x42,0x3a,0x6b,0x28,0x54,0xfa,0x85,0x3d,0xba},
197                  {0x2b,0x79,0x0a,0x15,0x9b,0x9f,0x5e,0xca,0x4e,0xd4,0xac,0xe5,0xf3,0x73,0xa7,0x57},
198                  {0xaf,0x58,0xa8,0x50,0xf4,0xea,0xd6,0x74,0x4f,0xae,0xe9,0xd5,0xe7,0xe6,0xad,0xe8},
199                  {0x2c,0xd7,0x75,0x7a,0xeb,0x16,0x0b,0xf5,0x59,0xcb,0x5f,0xb0,0x9c,0xa9,0x51,0xa0},
200                  {0x7f,0x0c,0xf6,0x6f,0x17,0xc4,0x49,0xec,0xd8,0x43,0x1f,0x2d,0xa4,0x76,0x7b,0xb7},
201                  {0xcc,0xbb,0x3e,0x5a,0xfb,0x60,0xb1,0x86,0x3b,0x52,0xa1,0x6c,0xaa,0x55,0x29,0x9d},
202                  {0x97,0xb2,0x87,0x90,0x61,0xbe,0xdc,0xfc,0xbc,0x95,0xcf,0xcd,0x37,0x3f,0x5b,0xd1},
203                  {0x53,0x39,0x84,0x3c,0x41,0xa2,0x6d,0x47,0x14,0x2a,0x9e,0x5d,0x56,0xf2,0xd3,0xab},
204                  {0x44,0x11,0x92,0xd9,0x23,0x20,0x2e,0x89,0xb4,0x7c,0xb8,0x26,0x77,0x99,0xe3,0xa5},
205                  {0x67,0x4a,0xed,0xde,0xc5,0x31,0xfe,0x18,0x0d,0x63,0x8c,0x80,0xc0,0xf7,0x70,0x07},
206                 };
207
208    B8 E[16][16]={{0x01,0x03,0x05,0x0f,0x11,0x33,0x55,0xff,0x1a,0x2e,0x72,0x96,0xa1,0xf8,0x13,0x35},
209                  {0x5f,0xe1,0x38,0x48,0xd8,0x73,0x95,0xa4,0xf7,0x02,0x06,0x0a,0x1e,0x22,0x66,0xaa},
210                  {0xe5,0x34,0x5c,0xe4,0x37,0x59,0xeb,0x26,0x6a,0xbe,0xd9,0x70,0x90,0xab,0xe6,0x31},
211                  {0x53,0xf5,0x04,0x0c,0x14,0x3c,0x44,0xcc,0x4f,0xd1,0x68,0xb8,0xd3,0x6e,0xb2,0xcd},
212                  {0x4c,0xd4,0x67,0xa9,0xe0,0x3b,0x4d,0xd7,0x62,0xa6,0xf1,0x08,0x18,0x28,0x78,0x88},
```

```
                {0xfe,0x19,0x2b,0x7d,0x87,0x92,0xad,0xec,0x2f,0x71,0x93,0xae,0xe9,0x20,0x60,0xa0},
                {0xfb,0x16,0x3a,0x4e,0xd2,0x6d,0xb7,0xc2,0x5d,0xe7,0x32,0x56,0xfa,0x15,0x3f,0x41},
                {0xc3,0x5e,0xe2,0x3d,0x47,0xc9,0x40,0xc0,0x5b,0xed,0x2c,0x74,0x9c,0xbf,0xda,0x75},
                {0x9f,0xba,0xd5,0x64,0xac,0xef,0x2a,0x7e,0x82,0x9d,0xbc,0xdf,0x7a,0x8e,0x89,0x80},
                {0x9b,0xb6,0xc1,0x58,0xe8,0x23,0x65,0xaf,0xea,0x25,0x6f,0xb1,0xc8,0x43,0xc5,0x54},
                {0xfc,0x1f,0x21,0x63,0xa5,0xf4,0x07,0x09,0x1b,0x2d,0x77,0x99,0xb0,0xcb,0x46,0xca},
                {0x45,0xcf,0x4a,0xde,0x79,0x8b,0x86,0x91,0xa8,0xe3,0x3e,0x42,0xc6,0x51,0xf3,0x0e},
                {0x12,0x36,0x5a,0xee,0x29,0x7b,0x8d,0x8c,0x8f,0x8a,0x85,0x94,0xa7,0xf2,0x0d,0x17},
                {0x39,0x4b,0xdd,0x7c,0x84,0x97,0xa2,0xfd,0x1c,0x24,0x6c,0xb4,0xc7,0x52,0xf6,0x01},
            };

    if(coef==0x02)
        celda=0x19+L[celda>>4][celda&(0x0f)];

    if(coef==0x03)
        celda=0x01+L[celda>>4][celda&(0x0f)];

    // xxxxxx("%x\n",L[xxlda>>4][xxlda&(0x0f)]);

    if(celda > 0xff)
        celda=celda-0xff;

    celda=E[celda>>4][celda&(0x0f)];

    return celda;

}
```

AddRoundKey: Se realizan los corrimientos pertinentes y las máscaras correspondientes para poder operar los valores adecuados, siendo la llave un vector de registros y la matriz de estado una matriz bidimensional, operando con una operación lógica de tipo XOR.

```
void AddRoundKey(B8 in[4][4], B32 w[Nb*(Nr+1)], int ronda){

    int i;

    for(i=0;i<4;i++){
        if(i==0){
            in[0][i]=(w[4*ronda]>>24) ^ in[0][i];
            in[1][i]=((w[4*ronda]>>16) & (0x00ff)) ^ in[1][i];
            in[2][i]=((w[4*ronda]>>8) & (0x0000ff)) ^ in[2][i];
            in[3][i]=(w[4*ronda] & (0x000000ff)) ^ in[3][i];

        }

        if(i==1){
            in[0][i]=(w[4*ronda+1]>>24) ^ in[0][i];
            in[1][i]=((w[4*ronda+1]>>16) & (0x00ff)) ^ in[1][i];
            in[2][i]=((w[4*ronda+1]>>8) & (0x0000ff)) ^ in[2][i];
            in[3][i]=(w[4*ronda+1] & (0x000000ff)) ^ in[3][i];

        }

        if(i==2){
            in[0][i]=(w[4*ronda+2]>>24) ^ in[0][i];
            in[1][i]=((w[4*ronda+2]>>16) & (0x00ff)) ^ in[1][i];
            in[2][i]=((w[4*ronda+2]>>8) & (0x0000ff)) ^ in[2][i];
            in[3][i]=(w[4*ronda+2] & (0x000000ff)) ^ in[3][i];

        }

        if(i==3){
            in[0][i]=(w[4*ronda+3]>>24) ^ in[0][i];
            in[1][i]=((w[4*ronda+3]>>16) & (0x00ff)) ^ in[1][i];
            in[2][i]=((w[4*ronda+3]>>8) & (0x0000ff)) ^ in[2][i];
            in[3][i]=(w[4*ronda+3] & (0x000000ff)) ^ in[3][i];
```

Salida del programa:



C:\Users\eco_b\OneDrive\Documentos\AES\cifradoAES.exe

```
Round[4].start  486c4eee671d9dd4de3b138d65f58e7
Round[4].s_box  52502f2885a45ed7e311c87f6cf6a94
Round[4].s_row  52a4c89485116a28e3cf2fd7f6505e7
Round[4].m_col  fd6daa9603138bf6fc0106b5eb3131
Round[4].k_sch  ef44a541a8525b7fb671253bdb0bad00
Round[5].start  e0927fe8c86363c0d9b1355085b8be1
Round[5].s_box  e14fd29be8fbfbba35c89653976cae7c
Round[5].s_row  e1fb967ce8c8ae9b356cd2ba974ffb53
Round[5].m_col  25d1a9adbd11d168b63a338e4c4cc0b0
Round[5].k_sch  d4d1c6f87c839d87caf2b8bc11f915bc
Round[6].start  f106f55c1924cef7cc88b325db5d5c
Round[6].s_box  a163a8fc784f29df10e83d234cd53fe
Round[6].s_row  a14f3dfe78e83fc10d5a8df4c632923
Round[6].m_col  4b868d6d2c4a8980339df4e837d218d8
Round[6].k_sch  6d88a37a110b3efddbf98641ca0093fd
Round[7].start  26e2e173d41b77de86472a9fdd28b25
Round[7].s_box  f7ab31f02783a9ff9b4340d354b53d3f
Round[7].s_row  f783403f27433df09bb531ff54aba9d3
Round[7].m_col  1415b5bf461615ec274656d7342ad843
Round[7].k_sch  4e54f70e5f5fc9f384a64fb24ea6dc4f
Round[8].start  5a4142b11949dc1fa3e019657a8c4c
Round[8].s_box  be832cc8d43b86c0ae1d44dda64f2fe
```

C:\Users\eco_b\OneDrive\Documentos\AES\cifradoAES.exe

```
Round[6].m_col  4b868d6d2c4a8980339df4e837d218d8
Round[6].k_sch  6d88a37a110b3efddbf98641ca0093fd
Round[7].start  26e2e173d41b77de86472a9fdd28b25
Round[7].s_box  f7ab31f02783a9ff9b4340d354b53d3f
Round[7].s_row  f783403f27433df09bb531ff54aba9d3
Round[7].m_col  1415b5bf461615ec274656d7342ad843
Round[7].k_sch  4e54f70e5f5fc9f384a64fb24ea6dc4f
Round[8].start  5a4142b11949dc1fa3e019657a8c4c
Round[8].s_box  be832cc8d43b86c0ae1d44dda64f2fe
Round[8].s_row  be3bd4fed4e1f2c8a642cc0da83864d
Round[8].m_col  0512fd1b1c889ff54766dcdfa1b99ea
Round[8].k_sch  ead27321b58dbad2312bf5607f8d292f
Round[9].start  ea835cf0445332d655d98ad8596b0c5
Round[9].s_box  87ec4a8cf26ec3d84d4c46959790e7a6
Round[9].s_row  876e46a6f24ce78c4d904ad897ecc395
Round[9].m_col  473794ed40d4e4a5a3703aa64c9f42bc
Round[9].k_sch  ac7766f319fadc2128d12941575c006e
Round[10].s_box e998972cb3175f3d327d94af2e2cb5
Round[10].s_row e9317db5cb322c723d2e895faf9794
Round[10].k_sch 5935807a7359f67f3d80477d4716fe3e
        OUT:  39  25  84  1d  2  dc  9  fb  dc  11  85  97  19  6a  b  32
```

Conclusiones:

Para esta práctica de cifrado en AES, logramos observar de una mejor manera como es que el cifrado AES realiza el proceso para poder cifrar la información del texto plano junto con la llave, ya que en cada uno de los pasos pudimos observar como es que se llevaban a cabo los puntos más importantes como el s_box, s_row y el mix columns, llegando a ello al resultado que se nos proporcionó en clase.

Banda Martínez César Eduardo

El algoritmo prevé una implementación de 10 rondas de cifrado debido a la naturaleza en la longitud de la llave, la cual es de 128 bits. El empleo de una rutina que permita el agilizar los productos polinomiales en la función MixColumns, repercute en un mayor control de la misma. Al ser AES un algoritmo de cifrado que opera sobre 2 campos finitos, resultó factible el dividir la problemática en el tratamiento de las operaciones sobre los ampos finitos siendo en MixColumns el campo GF(16) y en la función “FF” el campo GF(256).

Limón Hernández Raúl Rogelio