[CS304] Introduction to Cryptography and Network Security

Course Instructor: Dr. Dibyendu Roy — Winter 2023-2024
Scribed by: Dipean Dasgupta (202151188) — Lecture (Week 5)

1 Advanced Encryption Standard(AES):

One popular symmetric key encryption technique for protecting data is AES (Advanced Encryption Standard). It superseded the previous Data Encryption Standard (DES) when it was made a federal standard in 2001 by the National Institute of Standards and Technology (NIST). With support for key sizes of **128, 192, or 256 bits**, AES functions on data blocks.
Whereas DES employs a Feistel structure, AES functions more like a substitution-permutation network. The state, a 4-by-4 array of bytes, is changed repeatedly in rounds during the AES algorithm's execution. The cipher's input, which is 128 bits, or precisely 16 bytes, is what the state is originally set to. Next, in four phases every round, the state is subjected to the following operations:

- **S1: Add RoundKey**: The master key is used to generate a 128-bit sub-key, which is represented as a 4-by-4 array of bytes, for each AES round. This sub-key is used to XOR the state array, updating it.
- **S2: SubBytes:** In this stage, a single fixed lookup table S is used to replace each byte in the state array with a new byte. The S-box, often known as the substitution table, is a bijection over $\{0,1\}^8$
- **S3: ShiftRows:** During this phase, the bytes in every row of the state array are moved to the left in the following order: the array's first row remains unaltered, while the second, third, and fourth rows are moved one, two, and three places to the left, respectively. Every shift is cyclical, thus the first byte in the second row, for example, becomes the fourth byte.
- **S4: MixColumns**: The four bytes in each column undergo an invertible transformation in this stage. (In technical terms, this is a matrix multiplication over an appropriate field or a linear transformation.) This transformation's feature is that it produces two outputs that differ by at least 5 — b bytes if two inputs differ by b ¿ 0 bytes when applied.

AddRoundKey takes the place of MixColumns in the last round. This keeps an enemy from just flipping the final three steps, which are independent of the key. For this reason, despite being more difficult to develop, AES is now commonly utilized because it is far stronger than DES and triple DES.

1.1 Working Principles:

Instead of working with bits, AES works with bytes of data. The cipher processes 128 bits, or 16 bytes, of the incoming data at a moment because the block size is 128 bits. The key length determines how many rounds there are:

- 128 bit key – 10 rounds

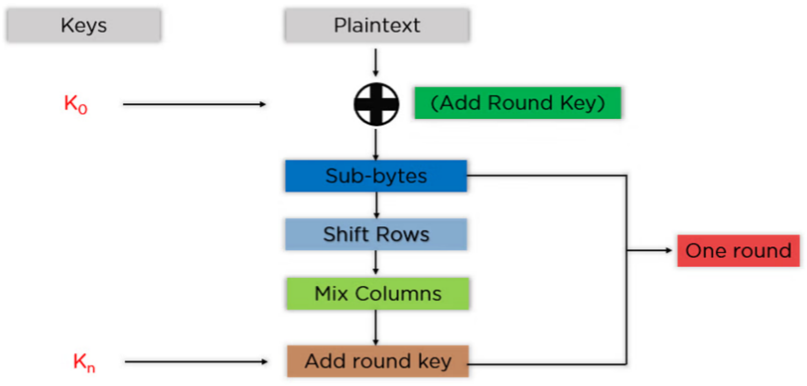


Figure:AES Encryption
Source:Simplilearn

- 192 bit key – 12 rounds
- 256 bit key – 14 rounds

1.1.1 Creating Round Keys:

To determine every round key from the key, an algorithm known as the Key Schedule is employed. To generate a variety of round keys for the associated encryption round, the beginning key is used. A sequence of round keys with the same length as the AES block size is produced by this technique. Round keys are produced by combining circular shifting of bytes, substitution via the S-box, and XOR operations. Every round of the encryption procedure uses these round keys, which increase security by causing confusion and dispersion. For particular operations, the last round of encryption uses the final round key.

AES-128 Key Scheduling Algorithm:
For creating the round keys for 128-bit encryption, 11 rounds are required. Let keys are:
$K_1, K_2 \dots K_{11}$
Length(each K_i) = 128 bits
$K = Key[0], Key[0] \dots Key[15]$
Length of Key[i] =128 bits

Rules:

- **ROTWORD**(B_0, B_1, B_2, B_3)=(B_1, B_2, B_3, B_0)
  performing left circular shift;
  Length of B_i=8 bits

- **SUBWORD**=(B_0, B_1, B_2, B_3)=$((B'_0, B'_1, B'_2, B'_3)$
  B'_i=subbytes(B_i)

A total of 10 constants are there for 11 rounds which are fixed.

RCon[1]=01000000	RCon[6]=20000000
RCon[2]=02000000	RCon[7]=40000000
RCon[3]=04000000	RCon[8]=80000000
RCon[4]=08000000	RCon[9]=1B000000
RCon[5]=10000000	RCon[10]=36000000

Constants for 11 rounds

Algorithm 1 AES KEY Scheduling Algorithm

```
Steps:
for $i = 0$ to 3 do
$w[i] \leftarrow (key[4i], key[4i+1], key[4i+2], key[4i+3])$
    for $i = 4$ to 43 do
        if $i \equiv 0 \pmod 4$ then $temp \leftarrow$ SUBWORD(ROTWORD($temp[i-1]$)) $\oplus RCon[i \div 4]$
        else
$temp \leftarrow temp[i-1]$
$w[i] \leftarrow w[i-4] \oplus temp$
return $w$
```

So, W[0] to W[43] each contain 32 bits.
In 44 blocks there are 44*32=1408 bits. Since each round has 128 bits,
Number of Rounds= 1408 ÷128 = **11 rounds.**
$Each of the 11 round$(K$_1, K_2 \dots K_{11}$) will have 4 blocks
$K_1 = w[0]||w[1]||w[2]||w[3] \dots$
$K_{11} = w[40]||w[41]||w[42]||w[43]$

1.1.2 ENCRYPTION:

Every block is viewed by AES as a 16-byte (4-byte x 4-byte = 128-byte) grid arranged in columns.

b0	b4	b8	b12
b1	b5	b9	b13
b2	b6	b10	b14
b3	b7	b11	b15

Each round comprises of 4 steps:

- **SubBytes**
- **ShiftRows**
- **MixColumns**
- **Add Round Key**

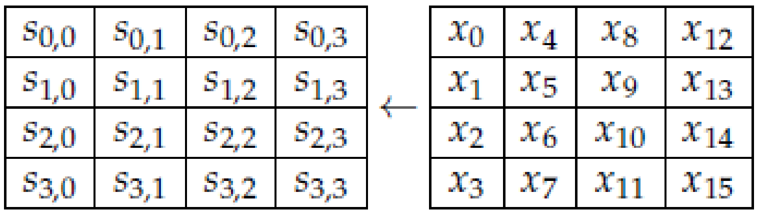
The MixColumns round is absent from the previous round. The algorithm's SubBytes handle substitution, while MixColumns and ShiftRows handle permutation.

NOTE: For the first 9 rounds, each round comprises of SubBytes, ShiftRows, MixColumns. From 10th round , each comprises of only SubBytes, ShiftRows.

SubBytes:
The substitution is put into practice in this stage. Every byte is changed to a different byte in this stage. The S-box, also known as a lookup table, is used to carry it out. The manner this replacement is carried out ensures that a byte is never replaced by itself or by a byte that is a compliment of the one that is being replaced.
As before, this step yields a 16 byte (4 x 4) matrix. The permutation is implemented in the next two phases.
The algorithm for subbytes are as follows:

```
Algorithm 2 SUBBYTES($a_7a_6a_5a_4a_3a_2a_1a_0$)
Require: External functions $FIELDINV$, $BINARYTOFIELD$, $FIELDTOBINARY$
Ensure: State $a$ is updated according to the SubBytes operation.
 1: $z \leftarrow BINARYTOFIELD(a_7a_6a_5a_4a_3a_2a_1a_0)$
 2: if $z \neq 0$ then
 3:     $z \leftarrow FIELDINV(z)$
 4: end if
 5: $(a_7a_6a_5a_4a_3a_2a_1a_0) \leftarrow FIELDTOBINARY(z)$
 6: $(C_7C_6C_5C_4C_3C_2C_1C_0) \leftarrow (01100011)$  ▷ In the following loop, all subscripts are to be reduced
    modulo 8
 7: for $i = 0$ to $7$ do
 8:     $b_i \leftarrow (a_i + a_{i+4} + a_{i+5} + a_{i+6} + a_{i+7} + C_i) \bmod 2$
 9: end for
10: return $(b_7b_6b_5b_4b_3b_2b_1b_0)$ =0
```

We now give precise descriptions of all the operations used in the AES; describe the structure of State; and discuss the construction of the key schedule. All operations in AES are byte-oriented operations, and all variables used are considered to be formed from an appropriate number of bytes. The plaintext x consists of 16 bytes, denoted $x_o \ldots x_{15}$. State is represented as a 4x4 array of bytes as follows:



$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$\leftarrow$

x_0	x_4	x_8	x_{12}
x_1	x_5	x_9	x_{13}
x_2	x_6	x_{10}	x_{14}
x_3	x_7	x_{11}	x_{15}

Figure: State representation
Source:GeeksForGeeks

The operation SUBBYTES performs a substitution on each byte of State independently, using an S-box, say π_s, which is a permutation of $\{0,1\}^8$.
To present this π_s, we represent bytes in hexadecimal notation. π_s is depicted as a 16 by 16 array, where the rows and columns are indexed by hexadecimal digits.
The permutation π_s incorporates operations in the finite field:
$F_{28} = \mathbb{Z}2[x]/(x^8 + x^2 + x^3 + x + 1)$.
Let FIELDINV denote the multiplicative inverse of a field element;let BINARY-TOFIELD convert a byte to a field element; and let FIELDTOBINARY perform the inverse conversion. This conversion is done in the obvious way: the field element:

$$\sum_{i=0}^{7} a_i x^i, \text{ where } a_i \in \mathbb{Z}_2 \text{ for } 0 \leq i \leq 7.$$

corresponds to the byte 7.Then the permutation ℓ_s is defined according to Algorithm. In this algorithm, the eight input bits $a_7a_6a_5a_4a_3a_2a_1a_0$ are replaced by the eight output bits $b_7b_6b_5b_4b_3b_2b_1b_0$.
For an example, let us begin with hexadecimal 53 In binary this is (01010011) which represents the field element

$$x^6 + x^2 + x + 1$$

The multiplicative inverse of this field F_{28},is:

$$x^7 + x^6 + x^3 + x$$

In binary notation, we have

$$(a_7a_6a_5a_4a_3a_2a_1a_0) = (00110110)$$

Next, we compute

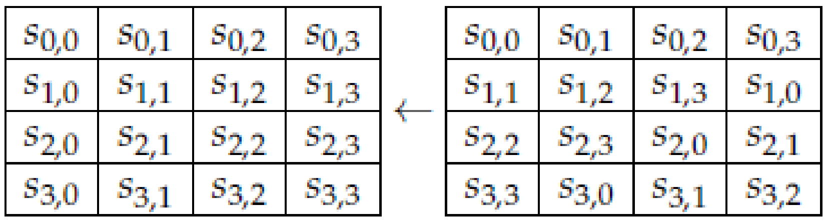
$$\begin{aligned} b_0 &= a_0 + a_4 + a_5 + a_6 + a_7 + c_0 mod 2 \\ &= 0 + 0 + 0 + 1 + 1 + 1 mod 2 \\ &= 1, \end{aligned}$$

followed by

$$\begin{aligned} b_1 &= a_1 + a_5 + a_6 + a_7 + c_0 mod 2 \\ &= 1 + 0 + 1 + 1 + 0 + 1 mod 2 \\ &= 0, \end{aligned}$$

In hexadecimal notation, 11101101 is ED.

ShiftRows:
This action is exactly what it sounds like. We move each row a certain amount of times. There is no shift to the top row. There is one leftward movement of the second row. There are two leftward shifts to the third row. There are three leftward shifts to the fourth row. (A circular shift to the left is executed.)



$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

$\longleftarrow$

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,1}$	$s_{1,2}$	$s_{1,3}$	$s_{1,0}$
$s_{2,2}$	$s_{2,3}$	$s_{2,0}$	$s_{2,1}$
$s_{3,3}$	$s_{3,0}$	$s_{3,1}$	$s_{3,2}$

Figure: Shifting of Rows
Source:GeeksForGeeks

MixColumns:
Essentially, matrix multiplication is what this step entails. Every column is multiplied by a particular matrix, which modifies each byte's location within the column. In the previous round, this step is omitted.
The relevant round key is now XORed with the output of the preceding step. In this case, the 16 bytes are simply regarded as 128 bits of data rather than a grid.

Following each of these rounds, 128 bits of encrypted data are returned as the output. Until all of the data that has to be encrypted has gone through this process, it is repeated.

2 Modes of Operation:

For DES, four modes of operation were created. In December 1980, FIPS Publication 81 included a standardization of them. When the block cipher is endomorphic, that is, when the plaintext(PT) and ciphertext(CT) spaces are the same, these modes of operation can be applied (with only slight modifications). Some other AES operating modes have been proposed more recently. The seven modes of operation listed below are well-known examples of modes, many of which are applied often in daily life.

- Electronic codebook mode (ECB mode)
- Cipher block chaining mode (CBC mode)
- Output feedback mode (OFB mode)
- Cipher feedback mode (CFB mode)
- Counter mode (CTR mode)
- Counter with cipher-block chaining MAC (CCM mode)
- Galois/counter mode (GCM)

2.1 ECB

Electronic codebook mode
Input: Key K n bit plaintext(PT) $x_i \dots x_t$
Encryption:

$$\text{EN}(x_i,\text{k})=C_i$$
$$1 \leq i \leq t \qquad C = C_1 \dots C_t$$

Decryption:

$$\text{DE}(C_i,\text{k})=x_i$$
$$1 \leq i \leq t$$

In real life, ECB mode is seldom ever utilized. The fact that identical plaintext blocks are encrypted into identical ciphertext(CT) blocks is a clear flaw in ECB mode. If the underlying message blocks are selected from a ”low entropy” plaintext(PT) space, then this is a major vulnerability. As an extreme example, ECB mode is effectively worthless if a plaintext(PT) block always includes wholly of 0s or totally of 1s.

2.2 CBC

Cipher Block Chaining
Input= K n bit PT Blocks $x_1, x_2 \dots X_t$
Encryption:

$$C_0 = IV$$
$$C_j = EN(C_{j-1}x_j, k)$$
$$1 \leq i \leq t$$

Decryption:

$$C_0 = IV$$
$$x_j = DE(C_j, k)C_{j-1}$$
$$1 \leq i \leq t$$

The encryption and decryption of CBC mode are presented graphically:

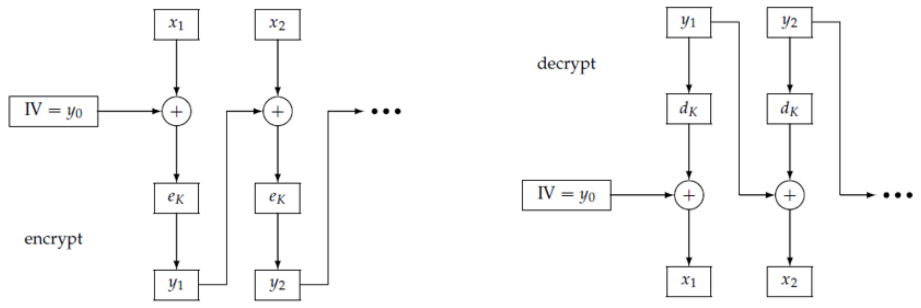


Figure: Encryption and Decryption in CBC mode
Source:Cryptography theory and practice

Although an IV is typically not secret, it is crucial to never use the same IV and key combination more than once when encrypting data. Therefore, an IV is usually selected with a suitable pseudo-random number generator and sent alongside the cipher text in an un-encrypted format.

3 Stream Ciphers:

One significant class of symmetric-key encryption systems is represented by stream ciphers. They are, in a way, extremely basic block ciphers with a block length of one. Their ability to alter the encryption transformation for every plain text(PT) symbol being encrypted is what makes them valuable. Because they do not propagate faults, stream ciphers are useful in scenarios where transmission failures are highly likely. They can also be utilized in situations when data processing needs to be done one symbol at a time (such as when the equipment has limited data buffering or no memory).
The encryption is done bitwise. (one bit at a time.)

$M = M_0 \dots m_l \quad m_i \in 0, 1$
$ENC(M, K) = e(M_0, Z_0), e(M_1, Z_1) \dots e(M_l, Z_l)$

$M = m_0 m_1 \dots m_l \quad m_i \in 0, 1$
$K = k_0 k_1 \dots k_l$
$C = m \oplus k = (m_0 \oplus k_0)(m_1 \oplus k_1) \dots (m_l \oplus k_l)$

Encryption: $C_i = m_i \oplus k_i$

Decryption: $M = C \oplus K$