**Advanced Encryption Standard (AES) Algorithm**

Advanced Encryption Standard (AES) is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001. AES is widely used today as it is much stronger than DES and triple DES despite being harder to implement.

**What is Advanced Encryption Standard (AES)?**

Advanced Encryption Standard (AES) is a highly trusted **encryption algorithm** used to secure data by converting it into an unreadable format without the proper key. Developed by the National Institute of Standards and Technology (NIST), **AES encryption** uses various **key lengths** (128, 192, or 256 bits) to provide strong protection against unauthorized access. This **data security** measure is efficient and widely implemented in securing **internet communication**, protecting **sensitive data**, and encrypting files. AES, a cornerstone of modern cryptography, is recognized globally for its ability to keep information safe from cyber threats.

AES is a Block Cipher, the key size can be 128/192/256 bits and encrypts data in blocks of 128 bits each.

That means it takes 128 bits as input and outputs 128 bits of encrypted cipher text. AES relies on the substitution-permutation network principle, which is performed using a series of linked operations that involve replacing and shuffling the input data.

**Working of The Cipher**

AES performs operations on bytes of data rather than in bits. Since the block size is 128 bits, the cipher processes 128 bits (or 16 bytes) of the input data at a time.

The number of rounds depends on the key length as follows :

* 128-bit key – 10 rounds
* 192-bit key – 12 rounds
* 256-bit key – 14 rounds

**Creation of Round Keys**

A Key Schedule algorithm calculates all the round keys from the key. So the initial key is used to create many different round keys which will be used in the corresponding round of the encryption.

**Encryption**

AES considers each block as a 16-byte (4 byte x 4 byte = 128 bits) grid in a column-major arrangement.

**Each round comprises of 4 steps:**

* SubBytes
* ShiftRows
* MixColumns
* Add Round Key

The last round doesn’t have the MixColumns round.

The SubBytes does the substitution and ShiftRows and MixColumns perform the permutation in the algorithm.

**Sub Bytes**

**This step implements the substitution.**

In this step, each byte is substituted by another byte. It is performed using a lookup table also called the S-box. This substitution is done in a way that a byte is never substituted by itself and also not substituted by another byte which is a compliment of the current byte. The result of this step is a 16-byte (4 x 4 ) matrix like before.

The next two steps implement the permutation.

**Shift Rows**

This step is just as it sounds. Each row is shifted a particular number of times.

* The first row is not shifted
* The second row is shifted once to the left.
* The third row is shifted twice to the left.
* The fourth row is shifted thrice to the left.

**Mix Columns**

This step is a matrix multiplication. Each column is multiplied with a specific matrix and thus the position of each byte in the column is changed as a result.

**Add Round Keys**

Now the resultant output of the previous stage is XOR-ed with the corresponding round key. Here, the 16 bytes are not considered as a grid but just as 128 bits of data.

After all these rounds 128 bits of encrypted data are given back as output. This process is repeated until all the data to be encrypted undergoes this process.

**Decryption**

The stages in the rounds can be easily undone as these stages have an opposite to it which when performed reverts the changes. Each 128 blocks goes through the 10,12 or 14 rounds depending on the key size.

The stages of each round of decryption are as follows :

* Add round key
* Inverse MixColumns
* ShiftRows
* Inverse SubByte

The decryption process is the encryption process done in reverse so I will explain the steps with notable differences.

**Inverse MixColumns**

 This step is similar to the Mix Columns step in encryption but differs in the matrix used to carry out the operation.

Mix Columns Operation each column is mixed independent of the other.

Matrix multiplication is used. The output of this step is the matrix multiplication of the old values and a constant matrix.

**Inverse SubBytes**

Inverse S-box is used as a lookup table and using which the bytes are substituted during decryption.

Function Substitute performs a byte substitution on each byte of the input word. For this purpose, it uses an S-box.