Advanced Encryption Standard (AES) Explainer

The Advanced Encryption Standard (AES) is a symmetric block cipher that has become the de facto standard for secure data encryption in numerous applications. Established by the U.S. National Institute of Standards and Technology (NIST) in 2001, AES was chosen after a rigorous selection process to replace the aging Data Encryption Standard (DES). AES operates on fixed-size blocks of 128 bits and supports key sizes of 128, 192, and 256 bits, providing a balance between security and performance that has made it suitable for a wide range of scenarios, from securing internet communications to protecting classified government information.

Key Features

1. Block size: 128 bits (16 bytes)
2. Key sizes: 128, 192, or 256 bits
3. Number of rounds:
   - 10 rounds for 128-bit keys
   - 12 rounds for 192-bit keys
   - 14 rounds for 256-bit keys

The varying number of rounds for different key sizes is a crucial aspect of AES's design, ensuring that the algorithm maintains its security properties across different key lengths. This adaptability allows AES to scale its computational complexity with the desired security level, making it versatile for various applications with different security requirements.

AES Structure

AES processes data in a 4x4 array of bytes called the "state". This structure is fundamental to understanding how AES manipulates data throughout its various transformations.

State Array Diagram

The state array can be represented mathematically as follows:

$$\text{state} = \begin{pmatrix} a_{0,0} & a_{0,1} & a_{0,2} & a_{0,3} \\ a_{1,0} & a_{1,1} & a_{1,2} & a_{1,3} \\ a_{2,0} & a_{2,1} & a_{2,2} & a_{2,3} \\ a_{3,0} & a_{3,1} & a_{3,2} & a_{3,3} \end{pmatrix}$$

Where each $a_{i,j}$ represents a byte (8 bits) of data.

In AES, the input plaintext is mapped to this state array column by column. For instance, a 128-bit (16-byte) plaintext block $p_0, p_1, \ldots, p_{15}$ would be mapped to the state as follows:

$$\text{state} = \begin{pmatrix} p_0 & p_4 & p_8 & p_{12} \\ p_1 & p_5 & p_9 & p_{13} \\ p_2 & p_6 & p_{10} & p_{14} \\ p_3 & p_7 & p_{11} & p_{15} \end{pmatrix}$$

This unique arrangement facilitates efficient implementation on various platforms and allows for effective diffusion of changes throughout the state during encryption.

Key Expansion

The key expansion process in AES is a critical component that generates a series of round keys from the original cipher key. This process ensures that each round of the algorithm uses a different key, enhancing the overall security by making it harder for an attacker to deduce the original key or find patterns in the encryption process.

Key Expansion Process Diagram

```
Original Key
    |
    v
[Initial Round Key]
    |
    v
[RotWord] -> [SubWord] -> [XOR with Rcon]
    |
    v
[XOR with previous word]
    |
    v
[Subsequent Round Keys]
```

Key Expansion Process Algorithm

Let's define the following:

N: length of the key in 32-bit words (4 for AES-128, 6 for AES-192, 8 for AES-256)

$K_0, K_1, ..., K_{N-1}$: 32-bit words of the original key

R: number of round keys needed (11 for AES-128, 13 for AES-192, 15 for AES-256)

$W_0, W_1, ..., W_{4R-1}$: 32-bit words of the expanded key

We also define two helper functions:

$\mathrm{RotWord}$: performs a one-byte left circular shift

$\mathrm{RotWord}([b_0, b_1, b_2, b_3]) = [b_1, b_2, b_3, b_0]$

$\mathrm{SubWord}$: applies the AES S-box to each of the four bytes of the word

$\mathrm{SubWord}([b_0, b_1, b_2, b_3]) = [S(b_0), S(b_1), S(b_2), S(b_3)]$

The key expansion algorithm can be expressed as follows: For $i = 0 \text{ to } 4R - 1$:

$$W_i = \begin{cases} K_i & \text{if } i < N \\ W_{i-N} \oplus \mathrm{SubWord}(\mathrm{RotWord}(W_{i-1})) \oplus \mathrm{rcon}_{i/N} & \text{if } i \geq N \text{ and } i \equiv 0 (\operatorname{mod} N) \\ W(i - N) \oplus \mathrm{SubWord}(W_{i-1}) & \text{if } i \geq N, N > 6, \text{and } i \equiv 4 (\operatorname{mod} N) \\ W_{i-N} \oplus W_{i-1} & \text{otherwise} \end{cases}$$

Where $\oplus$ denotes bitwise XOR.

Round Constant (Rcon) The round constant rcon_i for round i of the key expansion is a 32-bit word defined as: $\mathrm{rcon}i = [\mathrm{rc}_i, 00_{16}, 00_{16}, 00_{16}]$ Where rc_i is an eight-bit value defined recursively as:

$$\text{rc}i = \begin{cases} 01_{16} & \text{if } i = 1 \\ 02 \cdot \text{rc}_{i-1} & \text{if } i > 1 \text{ and } \text{rc}(i-1) < 80_{16} \\ (02 \cdot \text{rc}(i-1)) \oplus 11B_{16} & \text{if } i > 1 \text{ and } \text{rc}(i-1) \geq 80(16) \end{cases}$$

Alternatively, rc_i can be expressed as:

$\text{rc}_i = x^{i-1}$

Where the bits of rc_i are treated as coefficients of a polynomial in the finite field $\text{GF}(2)\frac{[x]}{x^8+x^4+x^3+x+1}$.

AES uses up to rcon_{10} for AES-128, rcon_8 for AES-192, and rcon_7 for AES-256.

This key expansion process generates a unique 128-bit key for each round of AES, regardless of the original key size. The total number of round keys generated depends on the key size: 11 for AES-128, 13 for AES-192, and 15 for AES-256 (including the original key).

Main Operations

SubBytes

The SubBytes transformation is a non-linear byte substitution that operates independently on each byte of the state using a substitution table (S-box). This operation provides confusion in the cipher, making it resistant to differential and linear cryptanalysis.

Mathematically, for each byte b in the state:

$b' = \text{S-box}[b]$

The S-box is constructed by composing two transformations:
1. Taking the multiplicative inverse in the finite field $\text{GF}(2^8)$, with 0x00 mapping to itself.
2. Applying an affine transformation over $\text{GF}(2)$:

$b'_i = b_i + b_{i+4 \bmod 8} + b_{i+5 \bmod 8} + b_{i+6 \bmod 8} + b_{i+7 \bmod 8} + c_i$

Where b_i is the i-th bit of the byte, and c_i is the i-th bit of a constant byte 0x63.

ShiftRows

The ShiftRows step provides diffusion in the cipher by cyclically shifting the last three rows of the state. This ensures that the four bytes of one column are spread out over four different columns in the output.

Mathematically, for row i (0-indexed):

$\text{ShiftRow}_{i(a_{\text{i},0},a_{\text{i},1},a_{\text{i},2},a_{\text{i},3})} = \left(a_{\text{i},(0+\text{i}) \bmod 4}, a_{\text{i},(1+\text{i}) \bmod 4}, a_{\text{i},(2+\text{i}) \bmod 4}, a_{\text{i},(3+\text{i}) \bmod 4}\right)$

MixColumns

The MixColumns step operates on each column of the state, treating it as a polynomial over $\text{GF}(2^8)$ and multiplying it with a fixed polynomial $a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$.

For a column $[s_0, s_1, s_2, s_3]^T$, the operation can be expressed as matrix multiplication:

$$\begin{pmatrix} s'_0 \\ s'_1 \\ s'_2 \\ s'_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \times \begin{pmatrix} s_0 \\ s_1 \\ s_2 \\ s_3 \end{pmatrix}$$

Where multiplication and addition are performed in $\mathrm{GF}(2^8)$.

AddRoundKey

In the AddRoundKey step, a round key is combined with the state using bitwise XOR operation. This step provides the core security of AES by introducing the key material into the state.

Mathematically:

$\text{state}' = \text{state} + \text{round\_key}$

Where $+$ denotes bitwise XOR.

Security Considerations

AES's security stems from its design as a substitution-permutation network, which provides excellent confusion and diffusion properties. The multiple rounds ensure that each bit of the ciphertext depends on every bit of the plaintext and key in a complex, non-linear way.

The security of AES against various attacks can be analyzed as follows:

1. Brute-force attacks: With key sizes of 128, 192, or 256 bits, AES provides security levels of 2^{128}, 2^{192}, and 2^{256} respectively, which are considered computationally infeasible with current and foreseeable technology.
2. Differential and linear cryptanalysis: AES was designed to be resistant to these attacks. The best known attacks require a computational complexity greater than exhaustive key search.
3. Related-key attacks: While some theoretical attacks exist for reduced-round versions of AES, full AES remains secure when properly implemented.
4. Side-channel attacks: These are implementation-specific and can be mitigated with proper countermeasures such as constant-time implementations and masking techniques.

In practice, AES-128 is considered secure for most applications, while AES-256 provides an extra margin of security for highly sensitive data or long-term protection against potential future advances in cryptanalysis or quantum computing.