Investigation of the RIJNDAEL Encryption Algorithm and Avalanche Effect

Advanced Encryption Standard

The Advanced Encryption Standard (AES) is a widely used symmetric block cipher designed for secure data encryption. It was established by the National Institute of Standards and Technology (NIST) in 2001 as a replacement for the older Data Encryption Standard (DES), which had become vulnerable to brute-force attacks. AES is now the standard encryption algorithm used in various security applications, including TLS, VPNs, disk encryption, and secure communications.

AES operates on 128-bit data blocks and supports three different key sizes: 128, 192, or 256 bits. Depending on the key length, the encryption process consists of 10 rounds (AES-128), 12 rounds (AES-192), or 14 rounds (AES-256). Each round of encryption applies a series of transformations designed to ensure strong diffusion and confusion, making it highly resistant to attacks.

The encryption process in AES consists of four main steps repeated across multiple rounds. The SubBytes step replaces each byte in the state matrix with a value from a predefined S-box, ensuring non-linearity. The ShiftRows step shifts the rows of the matrix to enhance diffusion. The MixColumns step further spreads the input data by mixing column values mathematically. Finally, the AddRoundKey step XORs the state with a round key, which is derived from the original encryption key using a key expansion algorithm. In the final round, the MixColumns step is omitted to ensure correct decryption.

One of AES's key strengths is its strong avalanche effect, meaning that even a minor change in the plaintext or key results in a significantly different ciphertext. This property makes AES highly secure against differential and linear cryptanalysis. Additionally, due to its efficient design, AES is both fast and optimized for software and hardware implementations, making it the preferred encryption standard for modern cybersecurity applications.

The Avalanche Effect in Cryptography

The avalanche effect is a fundamental property of cryptographic algorithms, particularly in block ciphers like AES and hash functions like SHA. It states that a small change in input (such as flipping a single bit in the plaintext or key) should cause a significant and unpredictable change in the output (ciphertext or hash). Ideally, about 50% of the output bits should change when a single input bit is modified.

This property is crucial for security because it ensures that attackers cannot predict how small input changes will affect the output, making cryptanalysis (such as differential cryptanalysis) extremely difficult.

Importance of the Avalanche Effect on Cryptography

a. Enhances Security
- Prevents attackers from predicting how input changes affect output, making brute-force and differential attacks ineffective.
- Ensures that even small changes in plaintext or key produce completely different ciphertexts.

b. Essential for Hash Functions
- Cryptographic hash functions like SHA-256 also rely on the avalanche effect.
- A slight input modification should result in a drastically different hash output (prevents collisions and ensures data integrity).

c. Protects Against Cryptanalysis
- Differential cryptanalysis relies on studying input-output relations, but the avalanche effect disrupts predictable patterns.
- Ensures that finding a relationship between plaintext, key, and ciphertext is computationally infeasible.

How the Avalanche Effect Works in AES

AES (Advanced Encryption Standard) is a block cipher that operates on 128-bit data blocks. It achieves strong diffusion through multiple encryption rounds, ensuring that any small input modification spreads widely across the ciphertext.

When encrypting a message, AES applies a series of transformations, including:

1. Byte Substitution (SubBytes) – Each byte is replaced using a nonlinear S-box.
2. Row Shifting (ShiftRows) – Rows of the state matrix are shifted.
3. Column Mixing (MixColumns) – Data is mixed across columns for diffusion.
4. Key Addition (AddRoundKey) – The round key is XORed with the state.

Each round increases the diffusion effect, making even a minor input change affect a growing number of ciphertext bits. By the final round (10th round in AES-128), nearly half of the ciphertext bits should change for a single-bit modification in the plaintext or key.

Implementation Example

Plain text: 128-bits

Two One Nine Two

54 77 6F 4F 6E 65 4E 69 6E 65 54 77 6F

Key: 128-bits

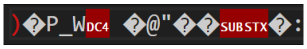
Thats my Kung Fu

54 68 61 74 73 6D 79 4B 75 6E 67 46 75

Encrypted message in hex:

29 c3 50 5f 57 14 20 f6 40 22 99 b3 1a 2 d7 3a

Encrypted message



```
)�P_WDC4 �@"��SUBSTX�:
```

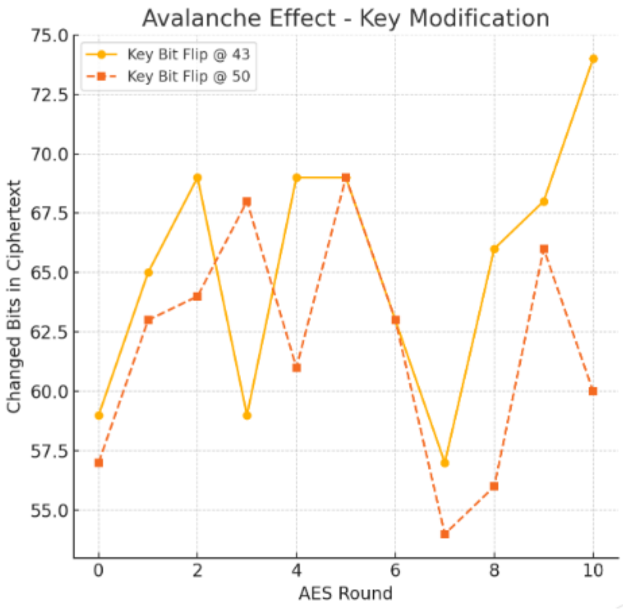
Decrypted message in hex:

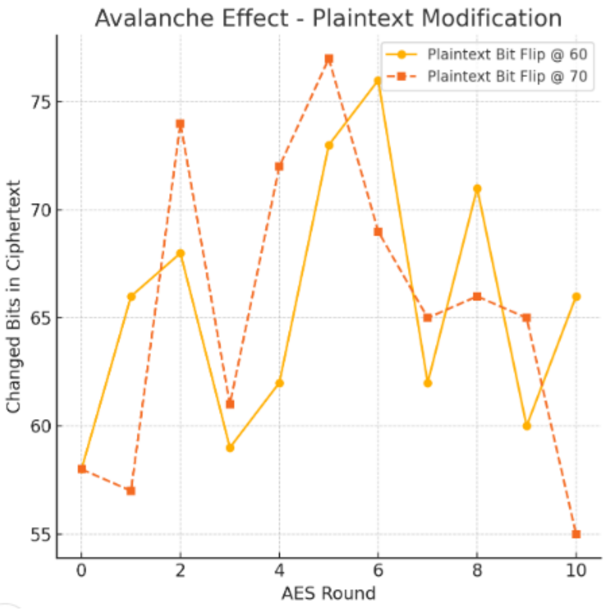
54 77 6f 20 4f 6e 65 20 4e 69 6e 65 20 54 77 6f 5b

Decrypted message:

Two One Nine Two[

Analysis & Conclusion





From the graphs, we can observe the following key insights about the avalanche effect in AES:

1. Diffusion Over Rounds

   - In both key and plaintext modifications, the number of changed bits increases rapidly in the first few rounds and stabilizes around round 5–6.
   - This aligns with AES's design, where diffusion spreads over multiple rounds.

2. Key vs. Plaintext Modifications
   - Key modifications (bit flips at positions 43 & 50) show a consistent pattern of changes, with some variations per round.
   - Plaintext modifications (bit flips at positions 60 & 70) result in a more fluctuating number of changed bits.
   - This suggests that altering the plaintext has a more irregular impact compared to altering the key, which affects the entire encryption process more uniformly.
3. Maximum and Minimum Diffusion
   - The maximum number of changed bits is around 76 (plaintext flip at round 6).
   - The minimum number of changed bits is around 54 (key flip at round 7).
   - Ideally, in a strong cryptographic algorithm like AES, half of the bits (~64 out of 128) should change on average.
4. Confirming the Avalanche Effect
   - Even a single bit flip in either the key or plaintext results in significant changes to the ciphertext.
   - This confirms the avalanche property, where small input changes cause drastic output changes.

Conclusion

In this study, we analyzed the avalanche effect in AES encryption by flipping a single bit in both the plaintext and the key to observe how the ciphertext changed. The results confirmed that even a minor change in the input leads to significant differences in the output, proving AES's strong diffusion properties.

A key takeaway from our analysis is that modifying the key caused a more uniform spread of changes across encryption rounds, whereas modifying the plaintext resulted in more fluctuation. However, in both cases, after about 5–6 rounds, the number of changed bits stabilized, aligning with AES's design to achieve full diffusion by the final round.

On average, 50–75 bits out of 128 changed in the ciphertext, which is consistent with the expected 50% bit change in a strong encryption algorithm. This confirms that AES is highly resistant to attacks like differential cryptanalysis since even a minor input change produces widespread and unpredictable alterations in the ciphertext.

From a practical standpoint, this experiment reinforces AES as a secure and reliable encryption standard, ensuring that encrypted data remains well-protected. The findings highlight the importance of using strong keys and properly structuring plaintext in cryptographic applications.

To further understand the workings of AES, implementing AES-128 in C++ would be a valuable next step. This implementation would include:

- Key Expansion: Generating round keys from the original 128-bit key.
- SubBytes Transformation: Applying an S-box substitution to enhance security.
- ShiftRows and MixColumns: Ensuring diffusion across the state matrix.
- AddRoundKey: Incorporating round keys into the encryption process.
- Finalization: Performing the last round without the MixColumns step.

By implementing AES-128 in C++, we can gain a deeper understanding of the encryption process and verify the correctness of the algorithm through testing. Additionally, this would allow us to conduct further experiments on different AES modes (e.g., CBC, CTR, GCM) to analyze their impact on security and performance.

Overall, this study confirmed the effectiveness of AES encryption and demonstrated how even a single-bit change can drastically alter the encrypted output. Future work could expand on this by implementing the algorithm in C++ and testing its behavior under different cryptographic conditions.