The Retracing Boomerang Attack

Gautam Singh

Indian Institute of Technology Hyderabad

April 28, 2025

Introduction

1. Broke the record for 5-round AES when it was published.

Introduction

1. Broke the record for 5-round AES when it was published.
2. Brings the attack complexity down to $2^{16.5}$ encryptions.

Introduction

1. Broke the record for 5-round AES when it was published.
2. Brings the attack complexity down to $2^{16.5}$ encryptions.
3. Uncovers a hidden relationship between boomerang attacks and two other cryptanalysis techniques: yoyo game and mixture differentials.

The Boomerang Attack

1. Typically split the encryption function as $E = E_1 \circ E_0$, with differential trails for each sub-cipher.

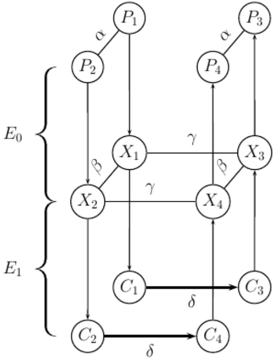

Figure 1: The boomerang attack.

The Boomerang Attack

1. Typically split the encryption function as $E = E_1 \circ E_0$, with differential trails for each sub-cipher.
2. We can build a distinguisher that can distinguish E from a truly random permutation in $\mathcal{O}((pq)^{-2})$ plaintext pairs.

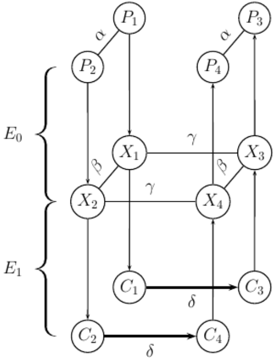

Figure 1: The boomerang attack.

The Boomerang Distinguisher

Algorithm 1 The Boomerang Attack Distinguisher

Initialize a counter $ctr \leftarrow 0$.
Generate $(pq)^{-2}$ plaintext pairs (P_1, P_2) such that $P_1 \oplus P_2 = \alpha$.
for all pairs (P_1, P_2) **do**
    Ask for the encryption of (P_1, P_2) to (C_1, C_2).
    Compute $C_3 = C_1 \oplus \delta$ and $C_4 = C_2 \oplus \delta$. ▷ *δ-shift*
    Ask for the decryption of (C_3, C_4) to (P_3, P_4).
    if $P_3 \oplus P_4 = \alpha$ **then**
        Increment *ctr*
if $ctr > 0$ **then**
    return This is the cipher E
else
    return This is a random permutation

Boomerang Switches

1. Gain 1-2 middle rounds for free by choosing differentials carefully. Here, we discuss the *S-box switch*.

Boomerang Switches

1. Gain 1-2 middle rounds for free by choosing differentials carefully. Here, we discuss the *S-box switch*.
2. Suppose the last operation in E_0 is a layer of S-boxes where $S(\rho_1\|\rho_2\|\ldots\|\rho_t) = (f_1(\rho_1)\|f_2(\rho_2)\|\ldots\|f_t(\rho_t))$ for t independent keyed functions f_i. Suppose the difference for both β and γ corresponding to the output of some f_j is equal to Δ.

Boomerang Switches

1. Gain 1-2 middle rounds for free by choosing differentials carefully. Here, we discuss the *S-box switch*.
2. Suppose the last operation in E_0 is a layer of S-boxes where $S(\rho_1\|\rho_2\|\ldots\|\rho_t) = (f_1(\rho_1)\|f_2(\rho_2)\|\ldots\|f_t(\rho_t))$ for t independent keyed functions f_i. Suppose the difference for both β and γ corresponding to the output of some f_j is equal to Δ.
3. Denoting this part of the intermediate state by X_j,

$$(X_1)_j \oplus (X_2)_j = (X_1)_j \oplus (X_3)_j = (X_2)_j \oplus (X_4)_j = \Delta \tag{1}$$

which shows $(X_1)_j = (X_4)_j$ and $(X_2)_j = (X_3)_j$.

Boomerang Switches

1. Gain 1-2 middle rounds for free by choosing differentials carefully. Here, we discuss the *S-box switch*.
2. Suppose the last operation in E_0 is a layer of S-boxes where $S(\rho_1 \| \rho_2 \| \ldots \| \rho_t) = (f_1(\rho_1) \| f_2(\rho_2) \| \ldots \| f_t(\rho_t))$ for t independent keyed functions f_i. Suppose the difference for both β and γ corresponding to the output of some f_j is equal to Δ.
3. Denoting this part of the intermediate state by X_j,
$$(X_1)_j \oplus (X_2)_j = (X_1)_j \oplus (X_3)_j = (X_2)_j \oplus (X_4)_j = \Delta \tag{1}$$
which shows $(X_1)_j = (X_4)_j$ and $(X_2)_j = (X_3)_j$.
4. If the differential characteristic in f_j^{-1} holds for (X_1, X_2), then it will hold for (X_3, X_4). *We pay for probability in one direction.*

Boomerang Switches

1. Gain 1-2 middle rounds for free by choosing differentials carefully. Here, we discuss the *S-box switch*.
2. Suppose the last operation in E_0 is a layer of S-boxes where $S(\rho_1 \| \rho_2 \| \dots \| \rho_t) = (f_1(\rho_1) \| f_2(\rho_2) \| \dots \| f_t(\rho_t))$ for t independent keyed functions f_i. Suppose the difference for both β and γ corresponding to the output of some f_j is equal to Δ.
3. Denoting this part of the intermediate state by X_j,

$$(X_1)_j \oplus (X_2)_j = (X_1)_j \oplus (X_3)_j = (X_2)_j \oplus (X_4)_j = \Delta \tag{1}$$

   which shows $(X_1)_j = (X_4)_j$ and $(X_2)_j = (X_3)_j$.
4. If the differential characteristic in f_j^{-1} holds for (X_1, X_2), then it will hold for (X_3, X_4). *We pay for probability in one direction.*
5. Distinguisher probability increases by a factor of $(q')^{-1}$, where q' is the probability of the differential characteristic in f_j.

The Yoyo Game

1. Similar to boomerang, starts by encrypting (P_1, P_2) to (C_1, C_2), then modifying them to (C_3, C_4) and decrypting them.

The Yoyo Game

1. Similar to boomerang, starts by encrypting (P_1, P_2) to (C_1, C_2), then modifying them to (C_3, C_4) and decrypting them.
2. *Unlike* the boomerang attack, this process continues in the yoyo game.

The Yoyo Game

1. Similar to boomerang, starts by encrypting (P_1, P_2) to (C_1, C_2), then modifying them to (C_3, C_4) and decrypting them.
2. *Unlike* the boomerang attack, this process continues in the yoyo game.
3. *All* pairs of intermediate values (X_{2l+1}, X_{2l+2}) satisfy some property (such as zero difference in some part).

The Yoyo Game

1. Similar to boomerang, starts by encrypting (P_1, P_2) to (C_1, C_2), then modifying them to (C_3, C_4) and decrypting them.
2. *Unlike* the boomerang attack, this process continues in the yoyo game.
3. *All* pairs of intermediate values (X_{2l+1}, X_{2l+2}) satisfy some property (such as zero difference in some part).
4. Probabilities are low with large l. Still, the yoyo technique has been used to attack AES reduced to 5 rounds.

Mixture

Definition 1 (Mixture)

Suppose $P_i \triangleq (\rho_1^i, \rho_2^i, \ldots, \rho_t^i)$. Given a plaintext pair (P_1, P_2), we say (P_3, P_4) is a *mixture counterpart* of (P_1, P_2) if for each $1 \leq j \leq t$, the quartet $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$ consists of two pairs of equal values or of four equal values. The quartet (P_1, P_2, P_3, P_4) is called a *mixture*.

Mixture

Definition 1 (Mixture)

Suppose $P_i \triangleq (\rho_1^i, \rho_2^i, \ldots, \rho_t^i)$. Given a plaintext pair (P_1, P_2), we say (P_3, P_4) is a *mixture counterpart* of (P_1, P_2) if for each $1 \leq j \leq t$, the quartet $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$ consists of two pairs of equal values or of four equal values. The quartet (P_1, P_2, P_3, P_4) is called a *mixture*.

1. If (P_1, P_2, P_3, P_4) is a mixture, then XOR of the intermediate values (X_1, X_2, X_3, X_4) is zero.

Mixture

Definition 1 (Mixture)

Suppose $P_i \triangleq (\rho_1^i, \rho_2^i, \ldots, \rho_t^i)$. Given a plaintext pair (P_1, P_2), we say (P_3, P_4) is a *mixture counterpart* of (P_1, P_2) if for each $1 \leq j \leq t$, the quartet $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$ consists of two pairs of equal values or of four equal values. The quartet (P_1, P_2, P_3, P_4) is called a *mixture*.

1. If (P_1, P_2, P_3, P_4) is a mixture, then XOR of the intermediate values (X_1, X_2, X_3, X_4) is zero.
2. $X_1 \oplus X_3 = \gamma \implies X_2 \oplus X_4 = \gamma$. Hence, for $\gamma \xrightarrow{q} \delta$ in E_1, $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ with probability q^2.

Mixture

Definition 1 (Mixture)

Suppose $P_i \triangleq (\rho_1^i, \rho_2^i, \ldots, \rho_t^i)$. Given a plaintext pair (P_1, P_2), we say (P_3, P_4) is a *mixture counterpart* of (P_1, P_2) if for each $1 \leq j \leq t$, the quartet $(\rho_j^1, \rho_j^2, \rho_j^3, \rho_j^4)$ consists of two pairs of equal values or of four equal values. The quartet (P_1, P_2, P_3, P_4) is called a *mixture*.

1. If (P_1, P_2, P_3, P_4) is a mixture, then XOR of the intermediate values (X_1, X_2, X_3, X_4) is zero.
2. $X_1 \oplus X_3 = \gamma \implies X_2 \oplus X_4 = \gamma$. Hence, for $\gamma \xrightarrow{q} \delta$ in E_1, $C_1 \oplus C_3 = C_2 \oplus C_4 = \delta$ with probability q^2.
3. Has been applied to AES reduced up to 6 rounds. E_0 is taken to be the first 1.5 rounds of AES, which can be treated as four parallel super S-boxes.

The Retracing Boomerang Framework

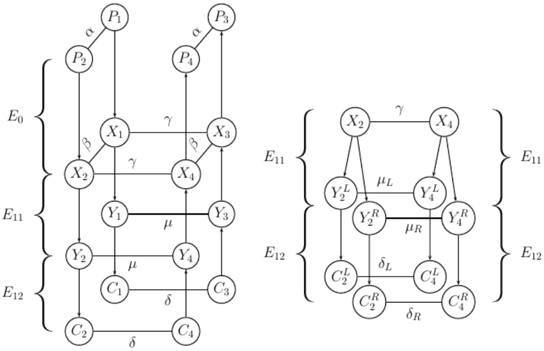

Figure 2: The retracing boomerang attack.

The Retracing Boomerang Attack

1. The *retracing boomerang* framework consists of a *shifting* type and a *mixing* type.

The Retracing Boomerang Attack

1. The *retracing boomerang* framework consists of a *shifting* type and a *mixing* type.
2. Both attacks use the setup shown in Figure 2.

The Retracing Boomerang Attack

1. The *retracing boomerang* framework consists of a *shifting* type and a *mixing* type.
2. Both attacks use the setup shown in Figure 2.
3. Although the additional split looks restrictive, it applies for a wide class of block ciphers such as SASAS constructions.

The Retracing Boomerang Attack

1. The *retracing boomerang* framework consists of a *shifting* type and a *mixing* type.
2. Both attacks use the setup shown in Figure 2.
3. Although the additional split looks restrictive, it applies for a wide class of block ciphers such as SASAS constructions.
4. Further, we assume that E_{12} can be split into two parts of size b and $n-b$ bits, call these functions E_{12}^{L} and E_{12}^{R}, with characteristic probabilities q_2^L and q_2^R respectively.

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation.*

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation*.
3. A δ-shift is performed on the filtered ciphertext pairs to get (C_3, C_4).

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation*.
3. A δ-shift is performed on the filtered ciphertext pairs to get (C_3, C_4).
4. Filtering ensures that the two unordered pairs (C_1, C_3) and (C_2, C_4) are *equal*.

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation*.
3. A δ-shift is performed on the filtered ciphertext pairs to get (C_3, C_4).
4. Filtering ensures that the two unordered pairs (C_1, C_3) and (C_2, C_4) are *equal*.
5. If one of these pairs satisfies the differential characteristic $\delta_L \xrightarrow{q_2^L} \mu_L$, *the other pair will too!*.

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation*.
3. A δ-shift is performed on the filtered ciphertext pairs to get (C_3, C_4).
4. Filtering ensures that the two unordered pairs (C_1, C_3) and (C_2, C_4) are *equal*.
5. If one of these pairs satisfies the differential characteristic $\delta_L \xrightarrow{q_2^L} \mu_L$, *the other pair will too!*.
6. Increases the probability of the boomerang distinguisher by $(q_2^L)^{-1}$.

The Shifting Retracing Boomerang Attack

1. Adds a $(b-1)$-bit filtering in the middle of the attack procedure.
2. Check if $C_1^L \oplus C_2^L = 0 \text{ or } \delta_L$. *Discard all such pairs that do not satisfy this relation*.
3. A δ-shift is performed on the filtered ciphertext pairs to get (C_3, C_4).
4. Filtering ensures that the two unordered pairs (C_1, C_3) and (C_2, C_4) are *equal*.
5. If one of these pairs satisfies the differential characteristic $\delta_L \xrightarrow{q_2^L} \mu_L$, *the other pair will too!*.
6. Increases the probability of the boomerang distinguisher by $(q_2^L)^{-1}$.
7. Any possible characteristic of (E_{12}^L) has probability at least 2^{-b+1}, thus the overall probability increases by a factor of at most 2^{b-1}. On the other hand, filtering only leaves 2^{-b+1} of the pairs, so there is no apparent gain.

The Shifting Retracing Boomerang Attack

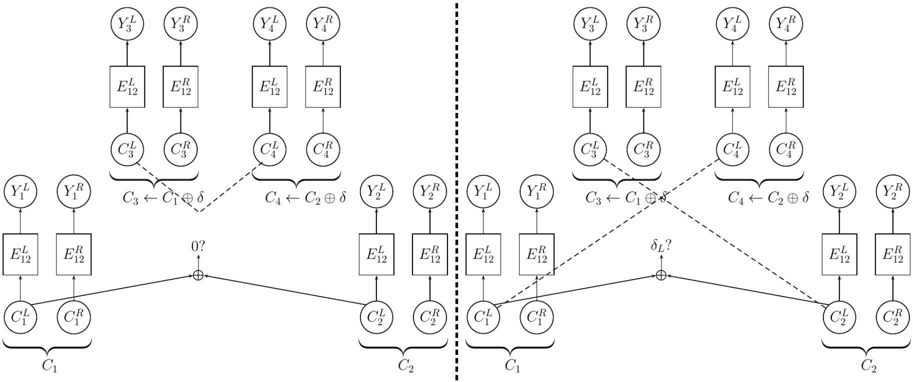

Figure 3: A shifted quartet (dashed lines indicate equality).

Advantages of Filtering

1. *Improving the signal to noise ratio.* Improving the probability by a factor of $(q_2^L)^{-1}$ improves the SNR which ensures a higher fraction of the filtered pairs on average satisfy $P_3 \oplus P_4 = \alpha$. The characteristic $\beta \xrightarrow{p} \alpha$ in the backward direction for the pair (X_3, X_4) can be replaced by a truncated differential characteristic $\beta \xrightarrow{p'} \alpha'$ of higher probability.

Advantages of Filtering

1. *Improving the signal to noise ratio.* Improving the probability by a factor of $(q_2^L)^{-1}$ improves the SNR which ensures a higher fraction of the filtered pairs on average satisfy $P_3 \oplus P_4 = \alpha$. The characteristic $\beta \xrightarrow{p} \alpha$ in the backward direction for the pair (X_3, X_4) can be replaced by a truncated differential characteristic $\beta \xrightarrow{p'} \alpha'$ of higher probability.
2. *Reducing the data complexity.* Due to the filtering, the attack leaves fewer ciphertexts. This improves the complexity in cases where more decryption queries are made.

Advantages of Filtering

1. *Improving the signal to noise ratio.* Improving the probability by a factor of $(q_2^L)^{-1}$ improves the SNR which ensures a higher fraction of the filtered pairs on average satisfy $P_3 \oplus P_4 = \alpha$. The characteristic $\beta \xrightarrow{p} \alpha$ in the backward direction for the pair (X_3, X_4) can be replaced by a truncated differential characteristic $\beta \xrightarrow{p'} \alpha'$ of higher probability.
2. *Reducing the data complexity.* Due to the filtering, the attack leaves fewer ciphertexts. This improves the complexity in cases where more decryption queries are made.
3. *Reducing the time complexity.* The filtering can also reduce the time complexity if it is dominated by the analysis of the plaintext pairs (P_3, P_4).

The Mixing Retracing Boomerang Attack

1. In the shifting attack, the attacker forces equality between the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) using a δ-shift.

The Mixing Retracing Boomerang Attack

1. In the shifting attack, the attacker forces equality between the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) using a δ-shift.
2. In this type of attack, each ciphertext pair can be shifted by $(C_1^L \oplus C_2^L, 0)$. The resulting ciphertexts are

$$C_3 = (C_3^L, C_3^R) = (C_1^L \oplus (C_1^L \oplus C_2^L), C_1^R) = (C_2^L, C_1^R), \tag{2}$$

$$C_4 = (C_4^L, C_4^R) = (C_2^L \oplus (C_1^L \oplus C_2^L), C_2^R) = (C_1^L, C_2^R). \tag{3}$$

The Mixing Retracing Boomerang Attack

1. In the shifting attack, the attacker forces equality between the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) using a δ-shift.
2. In this type of attack, each ciphertext pair can be shifted by $(C_1^L \oplus C_2^L, 0)$. The resulting ciphertexts are

$$C_3 = (C_3^L, C_3^R) = (C_1^L \oplus (C_1^L \oplus C_2^L), C_1^R) = (C_2^L, C_1^R), \tag{2}$$

$$C_4 = (C_4^L, C_4^R) = (C_2^L \oplus (C_1^L \oplus C_2^L), C_2^R) = (C_1^L, C_2^R). \tag{3}$$

3. Again, the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) are equal.

The Mixing Retracing Boomerang Attack

1. In the shifting attack, the attacker forces equality between the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) using a δ-shift.
2. In this type of attack, each ciphertext pair can be shifted by $(C_1^L \oplus C_2^L, 0)$. The resulting ciphertexts are

$$C_3 = (C_3^L, C_3^R) = (C_1^L \oplus (C_1^L \oplus C_2^L), C_1^R) = (C_2^L, C_1^R), \tag{2}$$
$$C_4 = (C_4^L, C_4^R) = (C_2^L \oplus (C_1^L \oplus C_2^L), C_2^R) = (C_1^L, C_2^R). \tag{3}$$

3. Again, the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) are equal.
4. Further, $C_1^R = C_3^R$ and $C_2^R = C_4^R$, thus we gain an *additional* factor of $(q_2^R)^{-2}$ for a total probability of $(pq_1)^2 q_2^L$, *better than shifting*!

The Mixing Retracing Boomerang Attack

1. In the shifting attack, the attacker forces equality between the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) using a δ-shift.
2. In this type of attack, each ciphertext pair can be shifted by $(C_1^L \oplus C_2^L, 0)$. The resulting ciphertexts are

$$C_3 = (C_3^L, C_3^R) = (C_1^L \oplus (C_1^L \oplus C_2^L), C_1^R) = (C_2^L, C_1^R), \quad (2)$$

$$C_4 = (C_4^L, C_4^R) = (C_2^L \oplus (C_1^L \oplus C_2^L), C_2^R) = (C_1^L, C_2^R). \quad (3)$$

3. Again, the unordered pairs (C_1^L, C_2^L) and (C_3^L, C_4^L) are equal.
4. Further, $C_1^R = C_3^R$ and $C_2^R = C_4^R$, thus we gain an *additional* factor of $(q_2^R)^{-2}$ for a total probability of $(pq_1)^2 q_2^L$, *better than shifting*!
5. Similar to the core step used in the yoyo attack on AES.

The Mixing Retracing Boomerang Attack

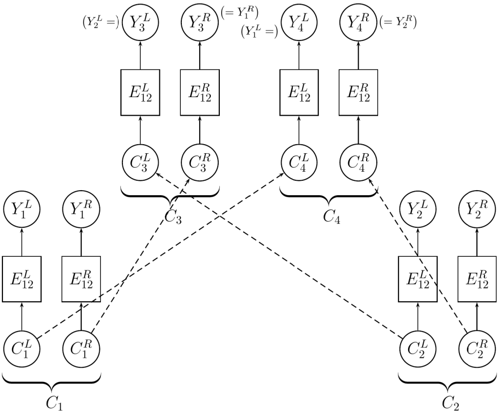

Figure 4: A mixture quartet of ciphertexts (dashed lines indicate equality).

Advantages of Shifting Retracing Attack

1. **Using structures**

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.

Advantages of Shifting Retracing Attack

1. **Using structures**
    - Shifting applies the same δ-shift to all pairs of ciphertexts.
    - Filtering is applied first to reduce the data complexity.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**
   - In mixing, the output difference of E_{12}^{L} is arbitrary.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**
   - In mixing, the output difference of E_{12}^L is arbitrary.
   - Usually no good combination between characteristics of $(E_{12}^L)^{-1}$ and $(E_{11})^{-1}$. For instance, in the yoyo attack, E_{11} is empty.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**
   - In mixing, the output difference of E_{12}^L is arbitrary.
   - Usually no good combination between characteristics of $(E_{12}^L)^{-1}$ and $(E_{11})^{-1}$. For instance, in the yoyo attack, E_{11} is empty.
3. **Construction of 'friend pairs'**

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**
   - In mixing, the output difference of E_{12}^L is arbitrary.
   - Usually no good combination between characteristics of $(E_{12}^L)^{-1}$ and $(E_{11})^{-1}$. For instance, in the yoyo attack, E_{11} is empty.
3. **Construction of 'friend pairs'**
   - 'Friend pairs' are pairs which satisfy a common property.

Advantages of Shifting Retracing Attack

1. **Using structures**
   - Shifting applies the same δ-shift to all pairs of ciphertexts.
   - Filtering is applied first to reduce the data complexity.
   - Not possible in mixing: shift is based on ciphertexts, no filtering.
   - Basic boomerang attacks add a round at the top or bottom of the distinguisher. With shifting, one can obtain all ciphertexts, shift them by δ and then decrypt, simulatneously checking for the filter and condition between P_3 and P_4 using a hash table.
2. **Combination with E_{11}**
   - In mixing, the output difference of E_{12}^L is arbitrary.
   - Usually no good combination between characteristics of $(E_{12}^L)^{-1}$ and $(E_{11})^{-1}$. For instance, in the yoyo attack, E_{11} is empty.
3. **Construction of 'friend pairs'**
   - 'Friend pairs' are pairs which satisfy a common property.
   - More 'friend pairs' can be constructed in the shifting variant.

Description of AES

1. Byte ordering shown after *SB* in Figure 5 (column major).

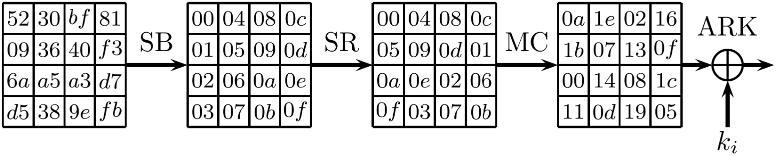

Figure 5: An AES round.

Description of AES

1. Byte ordering shown after *SB* in Figure 5 (column major).
2. j-th byte of a state X_i is denoted as $X_{i,j}$ or $(X_i)_j$.

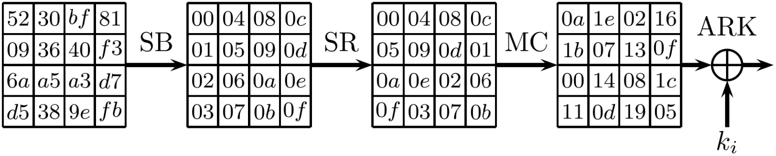

Figure 5: An AES round.

Description of AES

1. Byte ordering shown after *SB* in Figure 5 (column major).
2. j-th byte of a state X_i is denoted as $X_{i,j}$ or $(X_i)_j$.
3. Denote by W, Z and X the states before MC in round 0, at the input to round 1 and before MC in round 2 respectively.

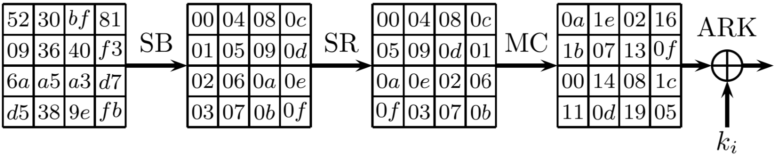

Figure 5: An AES round.

Description of AES

1. Byte ordering shown after SB in Figure 5 (column major).
2. j-th byte of a state X_i is denoted as $X_{i,j}$ or $(X_i)_j$.
3. Denote by W, Z and X the states before MC in round 0, at the input to round 1 and before MC in round 2 respectively.
4. The l-th shifted column (resp. l-th inverse shifted column) refers to application of SR (resp. SR^{-1}) to the l-th column.

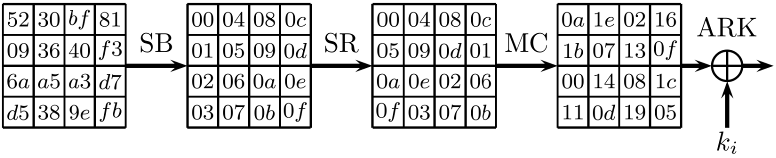

Figure 5: An AES round.

Description of AES

1. Byte ordering shown after SB in Figure 5 (column major).
2. j-th byte of a state X_i is denoted as $X_{i,j}$ or $(X_i)_j$.
3. Denote by W, Z and X the states before MC in round 0, at the input to round 1 and before MC in round 2 respectively.
4. The l-th shifted column (resp. l-th inverse shifted column) refers to application of SR (resp. SR^{-1}) to the l-th column.
5. Round subkeys are $k_{-1}, k_0, \ldots$.

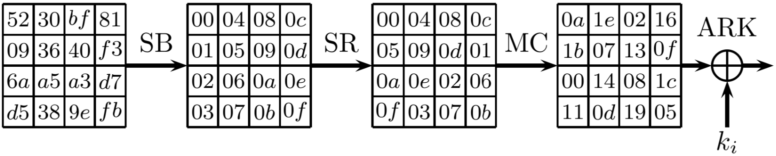

Figure 5: An AES round.

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.
2. Truncated differential characteristic for E_0: zero input difference in three inverse shifted columns and zero output difference in a single shifted column with probability $4 \cdot 2^{-8} = 2^{-6}$. (*why?*)

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.
2. Truncated differential characteristic for E_0: zero input difference in three inverse shifted columns and zero output difference in a single shifted column with probability $4 \cdot 2^{-8} = 2^{-6}$. (*why?*)
3. For E_{12}, 1.5 rounds of AES can be taken as four 32-bit super S-boxes.

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.
2. Truncated differential characteristic for E_0: zero input difference in three inverse shifted columns and zero output difference in a single shifted column with probability $4 \cdot 2^{-8} = 2^{-6}$. (*why?*)
3. For E_{12}, 1.5 rounds of AES can be taken as four 32-bit super S-boxes.
4. Ciphertext pair (C_1, C_2) modified into its mixture (C_3, C_4) w.r.t. super S-boxes and decrypted. The four inputs to the S-boxes have zero XOR, thus $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$ since MC is linear.

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.
2. Truncated differential characteristic for E_0: zero input difference in three inverse shifted columns and zero output difference in a single shifted column with probability $4 \cdot 2^{-8} = 2^{-6}$. (*why?*)
3. For E_{12}, 1.5 rounds of AES can be taken as four 32-bit super S-boxes.
4. Ciphertext pair (C_1, C_2) modified into its mixture (C_3, C_4) w.r.t. super S-boxes and decrypted. The four inputs to the S-boxes have zero XOR, thus $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$ since MC is linear.
5. $X_3 \oplus X_4 = 0$ in a shifted column and $Z_3 \oplus Z_4 = 0$ in an inverse shifted column with probability 2^{-6}. This corresponds to one of the four quartets $(0, 5, 10, 15)$, $(1, 4, 11, 14)$, $(2, 5, 8, 13)$, $(3, 6, 9, 12)$.

Summary of Yoyo Attack on Five Round AES

1. Decomposes AES as $E = E_{12} \circ E_{11} \circ E_0$ where E_0 is the first 2.5 rounds, E_{11} is the MC of round 2 and E_{12} is the last 2 rounds.
2. Truncated differential characteristic for E_0: zero input difference in three inverse shifted columns and zero output difference in a single shifted column with probability $4 \cdot 2^{-8} = 2^{-6}$. (*why?*)
3. For E_{12}, 1.5 rounds of AES can be taken as four 32-bit super S-boxes.
4. Ciphertext pair (C_1, C_2) modified into its mixture (C_3, C_4) w.r.t. super S-boxes and decrypted. The four inputs to the S-boxes have zero XOR, thus $X_1 \oplus X_2 \oplus X_3 \oplus X_4 = 0$ since MC is linear.
5. $X_3 \oplus X_4 = 0$ in a shifted column and $Z_3 \oplus Z_4 = 0$ in an inverse shifted column with probability 2^{-6}. This corresponds to one of the four quartets $(0, 5, 10, 15)$, $(1, 4, 11, 14)$, $(2, 5, 8, 13)$, $(3, 6, 9, 12)$.
6. Attack quartets of k_{-1}. Friend pairs of (Z_3, Z_4) used to get more information.

Algortihm of Yoyo Attack

Algorithm 2 Yoyo Attack on Five Round AES

1: Ask for the encryption of 2^6 pairs (P_1, P_2) of chosen plaintexts with non-zero difference only in bytes $0, 5, 10, 15$.
2: **for** all corresponding ciphertext pairs (C_1, C_2) **do**
3: Let (C_3^j, C_4^j), $j = 1, 2, 3, 4$ be the mixture counterparts of the pair (C_1, C_2).
4: Ask for the decryption of the ciphertext pairs and consider the pairs (Z_3^j, Z_4^j).
5: **for all** $l \in \{0, 1, 2, 3\}$ **do**
6: Assume all four pairs (Z_3^j, Z_4^j) and the pair (Z_1, Z_2) have zero difference in byte l.
7: Use the assumption to extract bytes $0, 5, 10, 15$ of k_{-1}.
8: **if** a contradiction is reached **then**
9: Increment l
10: **if** $l > 3$ **then** Discard the pair
11: **else**
12: Using $Z_3^j \oplus Z_4^j = 0$ in the entire l-th inverse shifted column, attack the three remaining columns of round 0 (sequentially) and decude the rest of k_{-1}.

Meet in the Middle Improvement on Yoyo Attack

1. Data complexity of 2^9 and time complexity of 2^{40}. Careful analysis of round 0 can reduce the complexity down to 2^{31}.

Meet in the Middle Improvement on Yoyo Attack

1. Data complexity of 2^9 and time complexity of 2^{40}. Careful analysis of round 0 can reduce the complexity down to 2^{31}.
2. Can improve using a meet in the middle (MITM) attack on bytes 0, 5, 10 and 15 of k_{-1}.

Meet in the Middle Improvement on Yoyo Attack

1. Data complexity of 2^9 and time complexity of 2^{40}. Careful analysis of round 0 can reduce the complexity down to 2^{31}.
2. Can improve using a meet in the middle (MITM) attack on bytes 0, 5, 10 and 15 of k_{-1}.
3. Denote the value of byte m before MC operation of round 0 by W_m, and WLOG let $l = 0$. Then,

$$Z_0 = 02_x \cdot W_0 \oplus 03_x \cdot W_1 \oplus 01_x \cdot W_2 \oplus 01_x \cdot W_3. \tag{4}$$

Meet in the Middle Improvement on Yoyo Attack

1. Data complexity of 2^9 and time complexity of 2^{40}. Careful analysis of round 0 can reduce the complexity down to 2^{31}.
2. Can improve using a meet in the middle (MITM) attack on bytes 0, 5, 10 and 15 of k_{-1}.
3. Denote the value of byte m before MC operation of round 0 by W_m, and WLOG let $l = 0$. Then,

$$Z_0 = 02_x \cdot W_0 \oplus 03_x \cdot W_1 \oplus 01_x \cdot W_2 \oplus 01_x \cdot W_3. \tag{4}$$

4. The adversary guesses bytes 0, 5 of k_{-1} by computing the following for $j = 1, 2, 3$ and storing the concatenated 24-bit value in a hash table.

$$02_x \cdot ((W_3^j)_0 \oplus (W_4^j)_0) \oplus 03_x \cdot ((W_3^j)_1 \oplus (W_4^j)_1) \tag{5}$$

Meet in the Middle Improvement on Yoyo Attack

5. Similarly, the adversary does this for bytes 10, 15 of k_{-1}, computing

$$01_x \cdot ((W_3^j)_2 \oplus (W_4^j)_2) \oplus 01_x \cdot ((W_3^j)_3 \oplus (W_4^j)_3) \tag{6}$$

Meet in the Middle Improvement on Yoyo Attack

5. Similarly, the adversary does this for bytes 10, 15 of k_{-1}, computing

$$01_x \cdot ((W_3^j)_2 \oplus (W_4^j)_2) \oplus 01_x \cdot ((W_3^j)_3 \oplus (W_4^j)_3) \tag{6}$$

6. The adversary checks for a match in the table, which is equivalent to $(Z_3^j)_0 = (Z_4^j)_0$ for $j = 1, 2, 3$.

Meet in the Middle Improvement on Yoyo Attack

5. Similarly, the adversary does this for bytes 10, 15 of k_{-1}, computing

$$01_x \cdot ((W_3^j)_2 \oplus (W_4^j)_2) \oplus 01_x \cdot ((W_3^j)_3 \oplus (W_4^j)_3) \tag{6}$$

6. The adversary checks for a match in the table, which is equivalent to $(Z_3^j)_0 = (Z_4^j)_0$ for $j = 1, 2, 3$.
7. 24-bit filtering leaves 2^8 candidates for bytes 0, 5, 10, 15 of k_{-1}, checked using the conditions $(Z_3^4)_0 = (Z_4^4)_0$ and $(Z_1)_0 = (Z_2)_0$.

Meet in the Middle Improvement on Yoyo Attack

5. Similarly, the adversary does this for bytes 10, 15 of k_{-1}, computing

$$01_x \cdot ((W_3^j)_2 \oplus (W_4^j)_2) \oplus 01_x \cdot ((W_3^j)_3 \oplus (W_4^j)_3) \tag{6}$$

6. The adversary checks for a match in the table, which is equivalent to $(Z_3^j)_0 = (Z_4^j)_0$ for $j = 1, 2, 3$.
7. 24-bit filtering leaves 2^8 candidates for bytes 0, 5, 10, 15 of k_{-1}, checked using the conditions $(Z_3^4)_0 = (Z_4^4)_0$ and $(Z_1)_0 = (Z_2)_0$.
8. Although the data complexity looks like 2^{16}, the *dissection technique* can be used to maintain the memory at 2^9.

Meet in the Middle Improvement on Yoyo Attack

5. Similarly, the adversary does this for bytes 10, 15 of k_{-1}, computing

$$01_x \cdot ((W_3^j)_2 \oplus (W_4^j)_2) \oplus 01_x \cdot ((W_3^j)_3 \oplus (W_4^j)_3) \tag{6}$$

6. The adversary checks for a match in the table, which is equivalent to $(Z_3^j)_0 = (Z_4^j)_0$ for $j = 1, 2, 3$.
7. 24-bit filtering leaves 2^8 candidates for bytes 0, 5, 10, 15 of k_{-1}, checked using the conditions $(Z_3^4)_0 = (Z_4^4)_0$ and $(Z_1)_0 = (Z_2)_0$.
8. Although the data complexity looks like 2^{16}, the *dissection technique* can be used to maintain the memory at 2^9.
9. The time complexity is now reduced to $2^6 \cdot 4 \cdot 2^{16} = 2^{24}$ operations, which is roughly equivalent to less than 2^{23} encryptions.

Improving the MITM Attack

1. The MITM attack can be improved to a time complexity of $2^{16.5}$ at the expense of incereasing the data complexity to 2^{15}.

Improving the MITM Attack

1. The MITM attack can be improved to a time complexity of $2^{16.5}$ at the expense of incereasing the data complexity to 2^{15}.
2. The main idea is to reduce the number of possible key values to 2^8 instead of 2^{16} as described earlier. This is done in multiple steps.

Improving the MITM Attack

1. The MITM attack can be improved to a time complexity of $2^{16.5}$ at the expense of incereasing the data complexity to 2^{15}.
2. The main idea is to reduce the number of possible key values to 2^8 instead of 2^{16} as described earlier. This is done in multiple steps.
    - Specific choice of plaintexts based on DDT of AES S-boxes.

Improving the MITM Attack

1. The MITM attack can be improved to a time complexity of $2^{16.5}$ at the expense of incereasing the data complexity to 2^{15}.
2. The main idea is to reduce the number of possible key values to 2^8 instead of 2^{16} as described earlier. This is done in multiple steps.
   - Specific choice of plaintexts based on DDT of AES S-boxes.
   - Eliminating key bytes using friend pairs.

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

2. Constrain $(P_1)_5 \oplus (P_2)_5 = 01_x$ to detect right key bytes efficiently.

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

2. Constrain $(P_1)_5 \oplus (P_2)_5 = 01_x$ to detect right key bytes efficiently.
3. DDT row of AES S-box for input difference 01_x along with input pair(s) for each output difference computed and stored in memory.

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

2. Constrain $(P_1)_5 \oplus (P_2)_5 = 01_x$ to detect right key bytes efficiently.
3. DDT row of AES S-box for input difference 01_x along with input pair(s) for each output difference computed and stored in memory.
4. For each (P_1, P_2) and for each guess of $k_{-1,0}$, use (7) to compute the output difference of the SB operation in byte 5.

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

2. Constrain $(P_1)_5 \oplus (P_2)_5 = 01_x$ to detect right key bytes efficiently.
3. DDT row of AES S-box for input difference 01_x along with input pair(s) for each output difference computed and stored in memory.
4. For each (P_1, P_2) and for each guess of $k_{-1,0}$, use (7) to compute the output difference of the SB operation in byte 5.
5. We can lookup to find inputs that can lead to this difference and retrieve possible values of $k_{-1,5}$ that correspond to the guessed $k_{-1,0}$.

Specific Choice of Plaintexts

1. Choose plaintexts with non-zero difference *only in bytes 0 and 5.* Here, $(Z_1)_0 = (Z_2)_0$ leaves 2^8 candidates for $k_{-1,\{0,5\}}$, given by

$$02_x \cdot ((W_1)_0 \oplus (W_2)_0) \oplus 03_x \cdot ((W_1)_1 \oplus (W_2)_1) = 0. \tag{7}$$

2. Constrain $(P_1)_5 \oplus (P_2)_5 = 01_x$ to detect right key bytes efficiently.
3. DDT row of AES S-box for input difference 01_x along with input pair(s) for each output difference computed and stored in memory.
4. For each (P_1, P_2) and for each guess of $k_{-1,0}$, use (7) to compute the output difference of the SB operation in byte 5.
5. We can lookup to find inputs that can lead to this difference and retrieve possible values of $k_{-1,5}$ that correspond to the guessed $k_{-1,0}$.
6. Gives 2^8 values of $k_{-1,\{0,5\}}$ in about 2^8 simple operations per pair.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which

$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$

Assume WLOG that equality holds in byte 10.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which
$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$
Assume WLOG that equality holds in byte 10.
3. Then, (6) depends only on $k_{-1,15}$ and has only 2^8 possible values.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which

$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$

   Assume WLOG that equality holds in byte 10.
3. Then, (6) depends only on $k_{-1,15}$ and has only 2^8 possible values.
4. Requires 2^9 simple operations and leaves 2^8 candidates for $k_{-1,\{0,5,15\}}$.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which

$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$

   Assume WLOG that equality holds in byte 10.
3. Then, (6) depends only on $k_{-1,15}$ and has only 2^8 possible values.
4. Requires 2^9 simple operations and leaves 2^8 candidates for $k_{-1,\{0,5,15\}}$.
5. Similar MITM procedure followed with another friend pair to obtain the unique value of $k_{-1,\{0,5,10,15\}}$ by isolating $k_{-1,10}$.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which
$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$
Assume WLOG that equality holds in byte 10.
3. Then, (6) depends only on $k_{-1,15}$ and has only 2^8 possible values.
4. Requires 2^9 simple operations and leaves 2^8 candidates for $k_{-1,\{0,5,15\}}$.
5. Similar MITM procedure followed with another friend pair to obtain the unique value of $k_{-1,\{0,5,10,15\}}$ by isolating $k_{-1,10}$.
6. Perform 2^8 operations for each pair (P_1, P_2) and for each value of l. Total time complexity of about 2^{16} operations.

Eliminating Key Bytes Using Friend Pairs

1. To reduce the number of candidates for $k_{-1,\{10,15\}}$, the boomerang process is used to return multiple friend pairs (P_3^j, P_4^j).
2. In particular, we choose one such pair for which
$$(P_3^j)_{10} \oplus (P_4^j)_{10} = 0 \quad \text{or} \quad (P_3^j)_{15} \oplus (P_4^j)_{15} = 0. \tag{8}$$
Assume WLOG that equality holds in byte 10.
3. Then, (6) depends only on $k_{-1,15}$ and has only 2^8 possible values.
4. Requires 2^9 simple operations and leaves 2^8 candidates for $k_{-1,\{0,5,15\}}$.
5. Similar MITM procedure followed with another friend pair to obtain the unique value of $k_{-1,\{0,5,10,15\}}$ by isolating $k_{-1,10}$.
6. Perform 2^8 operations for each pair (P_1, P_2) and for each value of l. Total time complexity of about 2^{16} operations.
7. Each pair requires 2^7 friend pairs to find one that satisfies (8) with high probability. Total data complexity is increased to about 2^{15}.

Attack Algorithm

1. **Precomputation:** Compute DDT row of AES S-box for input difference 01_x, along with actual inputs for each output difference.

Attack Algorithm

1. **Precomputation:** Compute DDT row of AES S-box for input difference 01_x, along with actual inputs for each output difference.
2. **Online Phase:** Take 64 pairs (P_1, P_2) with $(P_1)_5 = 00_x$, $(P_2)_5 = 01_x$, $(P_1)_0 \neq (P_2)_0$ and all other corresponding bytes equal.

Attack Algorithm

1. **Precomputation:** Compute DDT row of AES S-box for input difference 01_x, along with actual inputs for each output difference.
2. **Online Phase:** Take 64 pairs (P_1, P_2) with $(P_1)_5 = 00_x$, $(P_2)_5 = 01_x$, $(P_1)_0 \neq (P_2)_0$ and all other corresponding bytes equal.
3. For each plaintext pair, create 2^7 friend pairs (P_1^j, P_2^j) such that for each j, $P_1^j \oplus P_2^j = P_1 \oplus P_2$ and $(P_1^j)_{\{0,5,10,15\}} = (P_1)_{\{0,5,10,15\}}$.

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
    1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
    1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.
    2. Use the boomerang process to obtain pairs (P_3, P_4) and (P_3^j, P_4^j).

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
   1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.
   2. Use the boomerang process to obtain pairs (P_3, P_4) and (P_3^j, P_4^j).
   3. Find a j for which (8) is satisfied. Perform an MITM attack on column 0 of round 0 using (P_3^j, P_4^j) to obtain 2^8 candidates for $k_{-1,\{0,5,15\}}$.

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
    1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.
    2. Use the boomerang process to obtain pairs (P_3, P_4) and (P_3^j, P_4^j).
    3. Find a j for which (8) is satisfied. Perform an MITM attack on column 0 of round 0 using (P_3^j, P_4^j) to obtain 2^8 candidates for $k_{-1,\{0,5,15\}}$.
    4. Perform another MITM attack on column 0 of round 0 using two plaintext pairs $(P_3^{j'}, P_4^{j'})$. This gives a possible value for $k_{-1,\{0,5,10,15\}}$.

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
   1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.
   2. Use the boomerang process to obtain pairs (P_3, P_4) and (P_3^j, P_4^j).
   3. Find a j for which (8) is satisfied. Perform an MITM attack on column 0 of round 0 using (P_3^j, P_4^j) to obtain 2^8 candidates for $k_{-1,\{0,5,15\}}$.
   4. Perform another MITM attack on column 0 of round 0 using two plaintext pairs $(P_3^{j'}, P_4^{j'})$. This gives a possible value for $k_{-1,\{0,5,10,15\}}$.
   5. If contradiction, go to the next value of l. If contradiction for all l, discard this pair and go to the next pair.

Attack Algorithm

4. For each plaintext pair (P_1, P_2) and for each $l \in \{0, 1, 2, 3\}$, do the following. ($l = 0$ taken below)
   1. Use (7) to compute and store all 2^8 candidates for $k_{-1,\{0,5\}}$ in a table.
   2. Use the boomerang process to obtain pairs (P_3, P_4) and (P_3^j, P_4^j).
   3. Find a j for which (8) is satisfied. Perform an MITM attack on column 0 of round 0 using (P_3^j, P_4^j) to obtain 2^8 candidates for $k_{-1,\{0,5,15\}}$.
   4. Perform another MITM attack on column 0 of round 0 using two plaintext pairs $(P_3^{j'}, P_4^{j'})$. This gives a possible value for $k_{-1,\{0,5,10,15\}}$.
   5. If contradiction, go to the next value of l. If contradiction for all l, discard this pair and go to the next pair.
5. Using a pair (P_1, P_2) for which no contradiction occurred, perform similar MITM attacks on columns 1, 2 and 3 of round 0 using the fact that $Z_3 \oplus Z_4$ equals 0 in the l-th inverse shifted column to recover the entire k_{-1}.

Attack Analysis

1. The attack succeeds if the data contains a pair that satisfies the truncated differential characteristic of E_0 and for one of the 'friend pairs' of that pair, the corresponding plaintext pair (P_3^j, P_4^j) has zero difference in either byte 10 or 15.

Attack Analysis

1. The attack succeeds if the data contains a pair that satisfies the truncated differential characteristic of E_0 and for one of the 'friend pairs' of that pair, the corresponding plaintext pair (P_3^j, P_4^j) has zero difference in either byte 10 or 15.
2. Increasing the number of initial pairs and friend pairs per initial pair boosts success probability. With 64 pairs and 128 friend pairs per initial pair, the probability of success is $(1 - e^{-1})^2 \approx 0.4$

Attack Analysis

1. The attack succeeds if the data contains a pair that satisfies the truncated differential characteristic of E_0 and for one of the 'friend pairs' of that pair, the corresponding plaintext pair (P_3^j, P_4^j) has zero difference in either byte 10 or 15.
2. Increasing the number of initial pairs and friend pairs per initial pair boosts success probability. With 64 pairs and 128 friend pairs per initial pair, the probability of success is $(1 - e^{-1})^2 \approx 0.4$
3. Another way to boost succees probability is to find other ways to cancel terms in (6). For instance, if there exist j, j' such that $\{(P_3^j)_{10}, (P_4^j)_{10}\} = \{(P_3^{j'})_{10}, (P_4^{j'})_{10}\}$, we can take the XOR of (6) to cancel the effect of $k_{-1,10}$, thus increasing the success probability even when there is no pair that satisfies (8).

Attack Analysis

4. Data complexity is $2 \cdot 2^6 \cdot 2^7 = 2^{14}$ chosen plaintexts and 2^{14} adaptively chosen ciphertexts.

Attack Analysis

4. Data complexity is $2 \cdot 2^6 \cdot 2^7 = 2^{14}$ chosen plaintexts and 2^{14} adaptively chosen ciphertexts.
5. Structures can reduce the data complexity to slightly above 2^{14} adaptively chosen ciphertexts and plaintexts, but also slightly reduces the success probability due to additional dependencies between analyzed pairs.

Attack Analysis

4. Data complexity is $2 \cdot 2^6 \cdot 2^7 = 2^{14}$ chosen plaintexts and 2^{14} adaptively chosen ciphertexts.
5. Structures can reduce the data complexity to slightly above 2^{14} adaptively chosen ciphertexts and plaintexts, but also slightly reduces the success probability due to additional dependencies between analyzed pairs.
6. Memory complexity of the attack remains at 2^9 128-bit memory cells, like the yoyo attack.

Attack Analysis

4. Data complexity is $2 \cdot 2^6 \cdot 2^7 = 2^{14}$ chosen plaintexts and 2^{14} adaptively chosen ciphertexts.
5. Structures can reduce the data complexity to slightly above 2^{14} adaptively chosen ciphertexts and plaintexts, but also slightly reduces the success probability due to additional dependencies between analyzed pairs.
6. Memory complexity of the attack remains at 2^9 128-bit memory cells, like the yoyo attack.
7. Time complexity is dominated by several MITM attacks that take 2^{16} operations each. Considering one AES operation to be equivalent to 80 S-box lookups and adding it to the number of queries gives us a total of $2^{16.5}$ encryptions.

Attack on Five Round AES with a Secret S-box

1. Retracing boomerang attack recovers the secret key without fully recovering the secret S-box (the S-box is recovered upto an affine transformation in $GF(2^8)$).

Attack on Five Round AES with a Secret S-box

1. Retracing boomerang attack recovers the secret key without fully recovering the secret S-box (the S-box is recovered upto an affine transformation in $GF(2^8)$).
2. The idea exploit the fact that with probability 2^{-6}, the pair (Z_3, Z_4) has zero difference in an inverse shifted column.

Attack on Five Round AES with a Secret S-box

1. Retracing boomerang attack recovers the secret key without fully recovering the secret S-box (the S-box is recovered upto an affine transformation in $GF(2^8)$).
2. The idea exploit the fact that with probability 2^{-6}, the pair (Z_3, Z_4) has zero difference in an inverse shifted column.
3. This observation does not depend on the specific structure of MC and SB operations, hence it can be applied to key-dependent variants as well.

Setting up a System of Linear Equations

1. Assume WLOG the retracing boomerang produces zero difference in byte 0 of state Z, or $(Z_3)_0 \oplus (Z_4)_0 = 0$. (4) can be rewritten as

$$0 = (Z_3)_0 \oplus (Z_4)_0 \tag{9}$$

$$\begin{aligned} &= 02_x \cdot ((W_3)_0 \oplus (W_4)_0) \oplus 03_x \cdot ((W_3)_1 \oplus (W_4)_1) \\ &\quad \oplus 01_x \cdot ((W_3)_2 \oplus (W_4)_2) \oplus 01_x \cdot ((W_3)_3 \oplus (W_4)_3). \end{aligned} \tag{10}$$

Setting up a System of Linear Equations

1. Assume WLOG the retracing boomerang produces zero difference in byte 0 of state Z, or $(Z_3)_0 \oplus (Z_4)_0 = 0$. (4) can be rewritten as

$$0 = (Z_3)_0 \oplus (Z_4)_0 \tag{9}$$

$$\begin{aligned} &= 02_x \cdot ((W_3)_0 \oplus (W_4)_0) \oplus 03_x \cdot ((W_3)_1 \oplus (W_4)_1) \\ &\quad \oplus 01_x \cdot ((W_3)_2 \oplus (W_4)_2) \oplus 01_x \cdot ((W_3)_3 \oplus (W_4)_3). \end{aligned} \tag{10}$$

2. Note that $(W_3)_j = SB(P_3 \oplus k_{-1,j'})$ for $j = 0, 1, 2, 3$ where $j' = SR^{-1}(j)$.

Setting up a System of Linear Equations

1. Assume WLOG the retracing boomerang produces zero difference in byte 0 of state Z, or $(Z_3)_0 \oplus (Z_4)_0 = 0$. (4) can be rewritten as

$$0 = (Z_3)_0 \oplus (Z_4)_0 \tag{9}$$

$$\begin{aligned} &= 02_x \cdot ((W_3)_0 \oplus (W_4)_0) \oplus 03_x \cdot ((W_3)_1 \oplus (W_4)_1) \\ &\quad \oplus 01_x \cdot ((W_3)_2 \oplus (W_4)_2) \oplus 01_x \cdot ((W_3)_3 \oplus (W_4)_3). \end{aligned} \tag{10}$$

2. Note that $(W_3)_j = SB(P_3 \oplus k_{-1,j'})$ for $j = 0, 1, 2, 3$ where $j' = SR^{-1}(j)$.
3. If we define $4 \cdot 256 = 1024$ variables $x_{m,j} = SB(m \oplus k_{-1,j'})$ for $m \in \mathbb{F}_q$ and $j = 0, 1, 2, 3$, then each plaintext pair P_1, P_2 which satisfies (10) provides a linear equation in the variables $x_{m,j}$.

Setting up a System of Linear Equations

4. To obtain many pairs, attach about 2^{10} friend pairs to each of the 2^6 original pairs (P_1, P_2).

Setting up a System of Linear Equations

4. To obtain many pairs, attach about 2^{10} friend pairs to each of the 2^6 original pairs (P_1, P_2).
5. For each original pair along with its friend pairs, perform the mixing retracing boomerang process to obtain a linear equation in the variables $x_{m,j}$. A few more friend pairs are taken for extra filtering of the original pairs.

Setting up a System of Linear Equations

4. To obtain many pairs, attach about 2^{10} friend pairs to each of the 2^6 original pairs (P_1, P_2).
5. For each original pair along with its friend pairs, perform the mixing retracing boomerang process to obtain a linear equation in the variables $x_{m,j}$. A few more friend pairs are taken for extra filtering of the original pairs.
6. Since differences are used (10), we can recover the S-box with an invertible linear transformation over $GF(2^8)$. That is, we can only obtain functions S_0, S_1, S_2, S_3 such that

$$S_j(x) = L_0(SB(x \oplus k_{-1,j'})), \tag{11}$$

for some unknown linear transformation L_0. Similar linear transformations L_t will be obtained for column t.

The secret key k_{-1} can be recovered despite not knowing the S-box in two steps.

First, for each j', we can recover $\bar{k_{j'}} = k_{-1,0} \oplus k_{-1,j'}$ as $\bar{k_{j'}}$ is the unique value of c such that $S_j(x) = S_0(x \oplus c)$ for all x. Similarly, we can recover each inverse shifted column of k_{-1} up to 2^8 possible values. This reduces the total number of candidates for k_{-1} to 2^{32}.

Second, the differences $k_{-1,0} \oplus k_{-1,j}$ for $j = 1, 2, 3$ can be found by taking several quartets of values (x_0, x_1, x_2, x_3) such that $\bigoplus_{i=0}^{3} S_0(x_i) = 0$. These quartets eliminate the effect of the difference between the linear transformations L_0 and L_j by finding the unique value of c_j such that $\bigoplus_{i=0}^{3} S_j(c_j \oplus x) = 0$. Thus, in about 2^{12} operations, we can determine the entire secret key k_{-1} upto the value of $k_{-1,0}$. These 2^8 possibilities can be exhaustively searched.

The data complexity of this attack is $2 \cdot 2^6 \cdot 2^{10} = 2^{17}$ chosen plaintexts and 2^{17} adaptively chosen ciphertexts. Using structures, the amount of chosen plaintexts can be reduced to 2^{14}, thus the overall data complexity is less than $2^{17.5}$ chosen plaintexts and adaptively chosen ciphertexts.

The time complexity is dominated by solving a system of 1034 equations in 1024 variables for each of the 2^6 pairs (P_1, P_2). Using an efficient algorithm such as the Method of the Four Russians, each solution takes about 2^{27} simple operations or approximately 2^{21} encryptions. Thus, the overall time complexity is 2^{29}.

The memory complexity is dominated by the memory required for solving the equations, which is about 2^{17} 128-bit blocks.

The equation solving step has to be applied 2^8 times since we do not know if a pair satisfies the boomerang property. To obtain this information in advance, we can use the five-round yoyo distinguisher. In this variant, the time complexity is dominated by the complexity of the yoyo distinguisher, which is $2^{25.8}$. The memory complexity is still 2^{17}.