Differential Cryptanalysis

Kostas Papagiannopoulos
University of Amsterdam
k.papagiannopoulos@uva.nl

Contents

Introduction

Introduction

- Differential cryptanalysis is one of the strongest cryptanalytic attacks and targets the cipher design directly
- Invented publicly by Biham and Shamir (1990)
- However, it was already known to the NSA, and it affected the DES sbox, changing the original IBM design (1977)

Introduction

- Differential cryptanalysis is one of the strongest cryptanalytic attacks and targets the cipher design directly
- Invented publicly by Biham and Shamir (1990)
- However, it was already known to the NSA, and it affected the DES sbox, changing the original IBM design (1977)

- Novel ciphers are designed to resist differential cryptanalysis
- This lecture will introduce the attack using custom ciphers of growing complexity (CipherOne, CipherTwo, CipherThree, CipherFour)

Introduction

Attack idea:

- Consider the following trivial cipher

$$C = P \oplus K$$

- Note that the key K is constant i.e. the cipher is not the one-time pad

Introduction

Attack idea:

- Consider the following trivial cipher

$$C = P \oplus K$$

- Note that the key K is constant i.e. the cipher is not the one-time pad
- Encrypting messages m_0, m_1 using the same key and XORing the ciphertexts c_0, c_1 results in the following

$$c_0 \oplus c_1 = (m_0 \oplus k) \oplus (m_1 \oplus k) = m_0 \oplus m_1$$

- Notice that computing the difference between ciphertexts allows us to ignore the key k

Introduction

Attack idea:

- Consider the following trivial cipher

$$C = P \oplus K$$

- Note that the key K is constant i.e. the cipher is not the one-time pad
- Encrypting messages m_0, m_1 using the same key and XORing the ciphertexts c_0, c_1 results in the following

$$c_0 \oplus c_1 = (m_0 \oplus k) \oplus (m_1 \oplus k) = m_0 \oplus m_1$$

- Notice that computing the difference between ciphertexts allows us to ignore the key k
- **The goal of differential cryptanalysis is to recover the secret key. We assume that the attacker has access to the plaintext and ciphertext.**

CipherOne

CipherOne

- CipherOne is a block cipher with blocksize of 4 bits and keysize of 8 bits
- **CipherOne encryption algorithm**

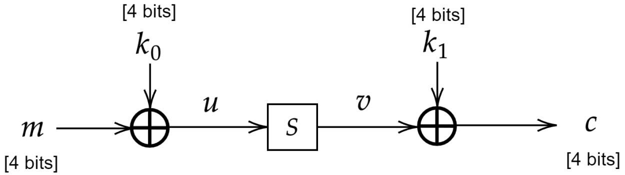
CipherOne($m, [k_0\ k_1]$)
$u = m \oplus k_0$
$v = S(u)$
$c = v \oplus k_1$

- It uses the following 4-bit sbox $S(\cdot)$, a 4-bit to 4-bit invertible function (popular in lightweight block ciphers)

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	6	4	c	5	0	7	2	e	1	f	3	d	8	a	9	b

CipherOne

- CipherOne with 4-bit input m, 8-bit key $[k_0 \;\; k_1]$ and 4-bit ciphertext c



- Encrypting two plaintexts messages m_0 and m_1 yields:

$$\begin{aligned} u_0 &= m_0 \oplus k_0 \\ v_0 &= S(u_0) \\ c_0 &= v_0 \oplus k_1 \end{aligned} \qquad \begin{aligned} u_1 &= m_1 \oplus k_0 \\ v_1 &= S(u_1) \\ c_1 &= v_1 \oplus k_1 \end{aligned}$$

CipherOne

Attack Algorithm:

1. **Link m to u.** Compute the difference between the intermediate values u_0, u_1

$$u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1$$

CipherOne

Attack Algorithm:

1. **Link m to u.** Compute the difference between the intermediate values u_0, u_1

$$u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1$$

2. **Link c to v.** Guess the value of the 4-bit key k_1 and for every guess $k_1 \in \{0, 1, \ldots, 15\}$ compute the intermediate values v_0, v_1

$$v_0 = k_1 \oplus c_0, \quad v_1 = k_1 \oplus c_1$$

CipherOne

Attack Algorithm:

1. **Link m to u.** Compute the difference between the intermediate values u_0, u_1

$$u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1$$

2. **Link c to v.** Guess the value of the 4-bit key k_1 and for every guess $k_1 \in \{0, 1, \ldots, 15\}$ compute the intermediate values v_0, v_1

$$v_0 = k_1 \oplus c_0, \quad v_1 = k_1 \oplus c_1$$

3. **Link v to u.** The 4-bit sbox is invertible, thus we can invert value v_0 to reach value u_0 and value v_1 to reach value u_1 (under certain key guess k_1)

$$u_0 = S^{-1}(v_0), \quad u_1 = S^{-1}(v_0)$$

CipherOne

Attack Algorithm:

1. **Link m to u.** Compute the difference between the intermediate values u_0, u_1

$$u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1$$

2. **Link c to v.** Guess the value of the 4-bit key k_1 and for every guess $k_1 \in \{0, 1, \ldots, 15\}$ compute the intermediate values v_0, v_1

$$v_0 = k_1 \oplus c_0, \quad v_1 = k_1 \oplus c_1$$

3. **Link v to u.** The 4-bit sbox is invertible, thus we can invert value v_0 to reach value u_0 and value v_1 to reach value u_1 (under certain key guess k_1)

$$u_0 = S^{-1}(v_0), \quad u_1 = S^{-1}(v_0)$$

4. If our key guess k_1 is correct, then it should hold that:

$$\boxed{m_0 \oplus m_1 = S^{-1}(v_0) \oplus S^{-1}(v_1)}$$

We refer to the formula above as the the **differential equation**

CipherOne

Generate n random 4-bit plaintext pairs with fixed difference, say `0x0f`

$m_0 \xleftarrow{R} \{0, 1, \ldots, 15\}$

$m_1 = m_0 \oplus \texttt{0x0f}$

$key = [0, 1, \ldots, 15]$

for $i = 1$ *until* n **do**

- $c_0 = CipherOne(m_0, [k_0\ k_1])$
- $c_1 = CipherOne(m_1, [k_0\ k_1])$
- $\delta_m = m_0 \oplus m_1 = \texttt{0x0f}$
- $candidates = \emptyset$
- **for** $k_1 = 0$ *until* 15 **do**
    - $u_0 = S^{-1}(k_1 \oplus c_0)$
    - $u_1 = S^{-1}(k_1 \oplus c_1)$
    - $\delta_u = u_0 \oplus u_1$
    - **if** $\delta_u == \delta_m$ **then**
        - $candidates = candidates \cup k_1$
    - **end**
- **end**
- **if** $candidates \neq \emptyset$ **then**
    - $key = candidates \cap key$
- **end**

end

CipherOne

- Having recovered the correct k_1, we can recover k_0 as well

$$c_0 = S(m_0 \oplus k_0) \oplus k_1 \iff k_0 = S^{-1}(c_0 \oplus k_1) \oplus m_0$$

- Differential cryptanalysis works by guessing parts of the key and testing whether the differential equation holds
- Verify the attack process using the MATLAB code in `dc_cipherone`

CipherTwo

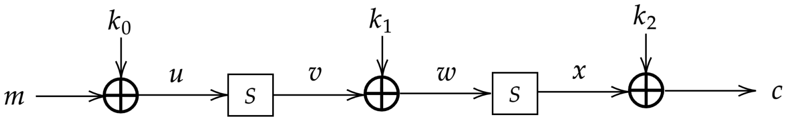
CipherTwo

- **CipherTwo encryption algorithm**

CipherTwo($m, [k_0\ k_1\ k_2]$)
$u = m \oplus k_0$
$v = S(u)$
$w = v \oplus k_1$
$x = S(w)$
$c = x \oplus k_2$

CipherTwo

- CipherTwo with 4-bit input m, 12-bit key $[k_0 \;\; k_1 \;\; k_2]$ and 4-bit ciphertext c



- Encrypting two plaintexts messages m_0 and m_1 yields:

$u_0 = m_0 \oplus k_0$	$u_1 = m_1 \oplus k_0$
$v_0 = S(u_0)$	$v_1 = S(u_1)$
$w_0 = v_0 \oplus k_1$	$w_1 = v_1 \oplus k_1$
$x_0 = S(w_0)$	$x_1 = S(w_1)$
$c_0 = x_0 \oplus k_2$	$c_1 = x_1 \oplus k_2$

CipherTwo

- **CipherTwo encryption algorithm**

CipherTwo($m, [k_0\ k_1\ k_2]$)
$u = m \oplus k_0$
$\mathbf{v} = \mathbf{S}(\mathbf{u})$
$w = v \oplus k_1$
$x = S(w)$
$c = x \oplus k_2$

- **Link c to x and link x to w.** If we guess the correct value of k_2 and invert the sbox, we can go backwards from the ciphertext pair c_0, c_1 to values w_0, w_1. The process is similar to CipherOne.

$$w_0 = S^{-1}(c_0 \oplus k_2), \quad w_1 = S^{-1}(c_1 \oplus k_2)$$

CipherTwo

- **CipherTwo encryption algorithm**

CipherTwo$(m, [k_0\ k_1\ k_2])$
$u = m \oplus k_0$
$\mathbf{v} = \mathbf{S}(\mathbf{u})$
$w = v \oplus k_1$
$x = S(w)$
$c = x \oplus k_2$

- **Link c to x and link x to w.** If we guess the correct value of k_2 and invert the sbox, we can go backwards from the ciphertext pair c_0, c_1 to values w_0, w_1. The process is similar to CipherOne.

$$w_0 = S^{-1}(c_0 \oplus k_2), \quad w_1 = S^{-1}(c_1 \oplus k_2)$$

- **Link w to v.** We can also link the difference δ_w (backwards) to the difference δ_v

$$\delta_w = w_0 \oplus w_1 = (v_0 \oplus k_1) \oplus (v_1 \oplus k_1) = v_0 \oplus v_1 = \delta_v$$

CipherTwo

- **CipherTwo encryption algorithm**

CipherTwo$(m, [k_0\ k_1\ k_2])$
$\mathbf{u} = \mathbf{m} \oplus \mathbf{k_0}$
$v = S(u)$
$w = v \oplus k_1$
$x = S(w)$
$c = x \oplus k_2$

- **Link m to u.** Going forward, we can link the difference δ_m to the difference δ_u (like CipherOne)

$$\delta_u = u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1 = \delta_m$$

CipherTwo

- **CipherTwo encryption algorithm**

CipherTwo$(m, [k_0\ k_1\ k_2])$
$\mathbf{u} = \mathbf{m} \oplus \mathbf{k_0}$
$v = S(u)$
$w = v \oplus k_1$
$x = S(w)$
$c = x \oplus k_2$

- **Link** m **to** u**.** Going forward, we can link the difference δ_m to the difference δ_u (like CipherOne)

$$\delta_u = u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1 = \delta_m$$

- The final task is to link the difference δ_u to the difference δ_v, yet this link is not directly visible

CipherTwo

- **CipherTwo encryption algorithm**

```
CipherTwo(m, [k0 k1 k2])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
c = x ⊕ k2
```

- **Link m to u.** Going forward, we can link the difference δ_m to the difference δ_u (like CipherOne)

$$\delta_u = u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1 = \delta_m$$

- The final task is to link the difference δ_u to the difference δ_v, yet this link is not directly visible
- Since $v = S(u)$, we will attempt to link the difference δ_v to the difference δ_u by analyzing the sbox

CipherTwo

- Consider all the sbox inputs u_0, u_1 such that their difference $\delta_u = u_0 \oplus u_1 = \texttt{0x0f}$
- Consider the respective sbox outputs $v_0 = S(u_0)$ and $v_1 = S(u_1)$

u_0	u_1	$\delta_u = u_0 \oplus u_1$	$v_0 = S(u_0)$	$v_1 = S(u_1)$	$\delta_v = v_0 \oplus v_1$
0	f	f	6	b	d
1	e	f	4	9	d
2	d	f	c	a	6
3	c	f	5	8	d
4	b	f	0	d	d
5	a	f	7	3	4
6	9	f	2	f	d
7	8	f	e	1	f
8	7	f	1	e	f
9	6	f	f	2	d
a	5	f	3	7	4
b	4	f	d	0	d
c	3	f	8	5	d
d	2	f	a	c	6
e	1	f	9	4	d
f	0	f	b	6	d

CipherTwo

- We observe that the distribution of δ_v when $\delta_u = \texttt{0x0f}$ is biased
- Not all values appear and certain values occur very frequently

  e.g. $Pr(\delta_v = \texttt{d}) = \dfrac{10}{16}$

u_0	u_1	$\delta_u = u_0 \oplus u_1$	$v_0 = S(u_0)$	$v_1 = S(u_1)$	$\delta_v = v_0 \oplus v_1$
0	f	f	6	b	d
1	e	f	4	9	d
2	d	f	c	a	6
3	c	f	5	8	d
4	b	f	0	d	d
5	a	f	7	3	4
6	9	f	2	f	d
7	8	f	e	1	f
8	7	f	1	e	f
9	6	f	f	2	d
a	5	f	3	7	4
b	4	f	d	0	d
c	3	f	8	5	d
d	2	f	a	c	6
e	1	f	9	4	d
f	0	f	b	6	d

CipherTwo

- To link the difference δ_u to the difference δ_v we will use the bias in the sbox output behavior
- We have shown that when $\delta_u = \texttt{0x0f}$, then δ_v is very likely (with probability $10/16$) to be equal to $\texttt{0x0d}$

CipherTwo

- To link the difference δ_u to the difference δ_v we will use the bias in the sbox output behavior
- We have shown that when $\delta_u = \texttt{0x0f}$, then δ_v is very likely (with probability $10/16$) to be equal to $\texttt{0x0d}$

- **Attack:** Note that differential cryptanalysis is a chosen-plaintext attack so we can generate plaintext pairs m_0, m_1 with difference:

$$\delta_m = m_0 \oplus m_1 = \texttt{0x0f}$$

- Since $\delta_m = \texttt{0x0f}$, then $\delta_u = \texttt{0x0f}$ and thus **it is likely** that $\delta_v = \texttt{0x0d}$

CipherTwo

- To link the difference δ_u to the difference δ_v we will use the bias in the sbox output behavior
- We have shown that when $\delta_u = \texttt{0x0f}$, then δ_v is very likely (with probability $10/16$) to be equal to $\texttt{0x0d}$

- **Attack:** Note that differential cryptanalysis is a chosen-plaintext attack so we can generate plaintext pairs m_0, m_1 with difference:

$$\delta_m = m_0 \oplus m_1 = \texttt{0x0f}$$

- Since $\delta_m = \texttt{0x0f}$, then $\delta_u = \texttt{0x0f}$ and thus **it is likely** that $\delta_v = \texttt{0x0d}$

- Starting from the ciphertext pair c_0, c_1 and guessing the key k_2 correctly will likely result in $\delta_v = \texttt{0x0d}$
- Starting from the ciphertext pair c_0, c_1 and guessing the key k_2 incorrectly is not likely to result in $\delta_v = \texttt{0x0d}$

$$\delta_v = \delta_w = S^{-1}(c_0 \oplus k_2) \oplus S^{-1}(c_1 \oplus k_2)$$

CipherTwo

Generate n random 4-bit plaintext pairs with fixed difference, say `0x0f`

$m_0 \xleftarrow{R} \{0, 1, \ldots, 15\}$

$m_1 = m_0 \oplus \texttt{0x0f}$

$counter(0 \text{ until } 15) = [0, 0, \ldots, 0]$

for $i = 1$ *until* n **do**

    $c_0 = CipherTwo(m_0, [k_0\ k_1\ k_2])$

    $c_1 = CipherTwo(m_1, [k_0\ k_1\ k_2])$

    for $k_2 = 0$ *until* 15 **do**

        $w_0 = S^{-1}(k_2 \oplus c_0)$

        $w_1 = S^{-1}(k_2 \oplus c_1)$

        $\delta_v = \delta_w = w_0 \oplus w_1$

        if $\delta_v == \mathit{0x0d}$ **then**

            $counter(k_2) = counter(k_2) + 1$

        end

    end

end

$key = argmax(counter)$

- Verify the attack process using the MATLAB code in `dc_ciphertwo`

CipherTwo

- We have used n plaintext/ciphertext pairs in the attack
- When we guess k_2 correctly, then $counter(k_2) = n \times \dfrac{10}{16}$ on average
- When we guess k_2 incorrectly, then $counter(k_2) = n \times \dfrac{1}{16}$ on average
- Thus the correct key is recovered by finding which key candidate has the highest counter value i.e. $argmax(counter)$

CipherThree

CipherThree

- We have seen in the cipher sbox $S(\cdot)$ that an input difference of `0x0f` leads to output difference of `0x0d` with probability 10/16
- Let's formalize this using the notion of **differential characteristic**

CipherThree

- We have seen in the cipher sbox $S(\cdot)$ that an input difference of `0x0f` leads to output difference of `0x0d` with probability 10/16
- Let's formalize this using the notion of **differential characteristic**

Differential characteristic. Let pair (α, β) such that the input difference α leads to output difference β. Assume this differential characteristic is associated with sbox $S(\cdot)$ and has probability p. We express the differential characteristic as follows:

$$\alpha \xrightarrow{S} \beta, \quad \text{with probability } p$$

CipherThree

- We have seen in the cipher sbox $S(\cdot)$ that an input difference of `0x0f` leads to output difference of `0x0d` with probability $10/16$
- Let's formalize this using the notion of **differential characteristic**

Differential characteristic. Let pair (α, β) such that the input difference α leads to output difference β. Assume this differential characteristic is associated with sbox $S(\cdot)$ and has probability p. We express the differential characteristic as follows:

$$\alpha \xrightarrow{S} \beta, \quad \text{with probability } p$$

$$\text{e.g.} \quad \texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

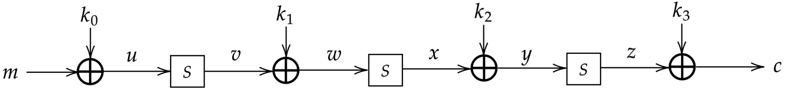
$$\text{e.g.} \quad \texttt{0x0f} \xrightarrow{S} \texttt{0x04}, \quad \text{with probability } 2/16$$

CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k0 k1 k2 k3])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
y = x ⊕ k2
z = S(y)
c = z ⊕ k3
```

- CipherThree with 4-bit input m, 16-bit key $[k_0 \;\; k_1 \;\; k_2 \;\; k_3]$ and 4-bit ciphertext c



CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k0 k1 k2 k3])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
y = x ⊕ k2
z = S(y)
c = z ⊕ k3
```

- **Link c to z and y.** Moving backwards, we can guess the correct value of k_3 and invert the sbox to reach the values y_0, y_1

$$y_0 = S^{-1}(c_0 \oplus k_3), \quad y_1 = S^{-1}(c_1 \oplus k_3)$$

CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k0 k1 k2 k3])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
y = x ⊕ k2
z = S(y)
c = z ⊕ k3
```

- **Link c to z and y.** Moving backwards, we can guess the correct value of k_3 and invert the sbox to reach the values y_0, y_1

$$y_0 = S^{-1}(c_0 \oplus k_3), \quad y_1 = S^{-1}(c_1 \oplus k_3)$$

- **Link y to x.** Moving backwards, we link the difference δ_y to the difference δ_x

$$\delta_y = y_0 \oplus y_1 = (x_0 \oplus k_2) \oplus (x_1 \oplus k_2) = x_0 \oplus x_1 = \delta_x$$

CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k0 k1 k2 k3])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
y = x ⊕ k2
z = S(y)
c = z ⊕ k3
```

- **Link c to z and y.** Moving backwards, we can guess the correct value of k_3 and invert the sbox to reach the values y_0, y_1

$$y_0 = S^{-1}(c_0 \oplus k_3), \quad y_1 = S^{-1}(c_1 \oplus k_3)$$

- **Link y to x.** Moving backwards, we link the difference δ_y to the difference δ_x

$$\delta_y = y_0 \oplus y_1 = (x_0 \oplus k_2) \oplus (x_1 \oplus k_2) = x_0 \oplus x_1 = \delta_x$$

- **Link m to u.** Moving forward, we link the difference δ_u to the plaintext difference δ_m

$$\delta_u = u_0 \oplus u_1 = (m_0 \oplus k_0) \oplus (m_1 \oplus k_0) = m_0 \oplus m_1 = \delta_m$$

CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k0 k1 k2 k3])
u = m ⊕ k0
v = S(u)
w = v ⊕ k1
x = S(w)
y = x ⊕ k2
z = S(y)
c = z ⊕ k3
```

- **Link u to v.** Moving forward, see that we have already linked the difference δ_u to δ_v using the differential characteristic:

$$\texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

CipherThree

- **CipherThree encryption algorithm**

CipherThree$(m, [k_0\ k_1\ k_2\ k_3])$
$u = m \oplus k_0$
$v = S(u)$
$w = v \oplus k_1$
$x = S(w)$
$y = x \oplus k_2$
$z = S(y)$
$c = z \oplus k_3$

- **Link u to v.** Moving forward, see that we have already linked the difference δ_u to δ_v using the differential characteristic:

$$\texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

- **Link v to w.** Moving forward, see that $\delta_w = \delta_v$

$$\delta_w = w_0 \oplus w_1 = (v_0 \oplus k_1) \oplus (v_1 \oplus k_1) = v_0 \oplus v_1 = \delta_v$$

- **Link y to x.** Moving backwards, see that $\delta_x = \delta_y$

$$\delta_y = y_0 \oplus y_1 = (x_0 \oplus k_2) \oplus (x_1 \oplus k_2) = x_0 \oplus x_1 = \delta_x$$

CipherThree

- **CipherThree encryption algorithm**

$\text{CipherThree}(m, [k_0\ k_1\ k_2\ k_3])$
$u = m \oplus k_0$
$v = S(u)$
$w = v \oplus k_1$
$x = S(w)$
$y = x \oplus k_2$
$z = S(y)$
$c = z \oplus k_3$

- The only remaining link to establish is between δ_w and δ_x
- We have shown that if $\delta_m = \texttt{0x0f}$ then $\delta_v = \texttt{0x0d}$ with probability $10/16$ and thus also that $\delta_w = \delta_v = \texttt{0x0d}$ with probability $10/16$

CipherThree

- **CipherThree encryption algorithm**

```
CipherThree(m, [k_0 k_1 k_2 k_3])
u = m ⊕ k_0
v = S(u)
w = v ⊕ k_1
x = S(w)
y = x ⊕ k_2
z = S(y)
c = z ⊕ k_3
```

- The only remaining link to establish is between δ_w and δ_x
- We have shown that if $\delta_m = \texttt{0x0f}$ then $\delta_v = \texttt{0x0d}$ with probability $10/16$ and thus also that $\delta_w = \delta_v = \texttt{0x0d}$ with probability $10/16$
- Since it is likely that the $\delta_w = \texttt{0x0d}$, we must apply yet another differential characteristic of the sbox

$$\texttt{0x0d} \xrightarrow{S} \texttt{?}$$

CipherThree

w_0	w_1	$\delta_w = w_0 \oplus w_1$	$x_0 = S(w_0)$	$x_1 = S(w_1)$	$\delta_x = x_0 \oplus x_1$
0	d	d	6	a	c
1	c	d	4	8	c
2	f	d	c	b	7
3	e	d	5	9	c
4	9	d	0	f	f
5	8	d	7	1	6
6	b	d	2	d	f
7	a	d	e	3	d
8	5	d	1	7	6
9	4	d	f	0	f
a	7	d	3	e	d
b	6	d	d	2	f
c	1	d	8	4	c
d	0	d	a	6	c
e	3	d	9	5	c
f	2	d	b	c	7

- Observe that the sbox output difference 0x0c appears frequently

  i.e. $Pr(\delta_v = \texttt{0x0c}) = \dfrac{6}{10}$
- We have found another useful differential characteristic for the sbox $S(\cdot)$

$$\texttt{0x0d} \xrightarrow{S} \texttt{0x0c}, \quad \text{with probability } 6/16$$

CipherThree

- The following links have been established:

$$\delta_u \text{ to } \delta_w: \quad \texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

$$\delta_w \text{ to } \delta_x: \quad \texttt{0x0d} \xrightarrow{S} \texttt{0x0c}, \quad \text{with probability } 6/16$$

CipherThree

- The following links have been established:

$$\delta_u \text{ to } \delta_w: \quad \texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

$$\delta_w \text{ to } \delta_x: \quad \texttt{0x0d} \xrightarrow{S} \texttt{0x0c}, \quad \text{with probability } 6/16$$

- Joining the two differential characteristics and assuming that they are independent we get:

$$\delta_u \text{ to } \delta_x: \quad \texttt{0x0f} \xrightarrow{S} \texttt{0x0d} \xrightarrow{S} \texttt{0x0c}$$

$$\text{with probability } 10/16 * 6/16 = 15/64$$

CipherThree

Generate n random 4-bit plaintext pairs with fixed difference, say 0x0f

$m_0 \xleftarrow{R} \{0, 1, \ldots, 15\}$

$m_1 = m_0 \oplus \texttt{0x0f}$

$counter(0 \text{ until } 15) = [0, 0, \ldots, 0]$

for $i = 1$ *until* n **do**

    $c_0 = CipherThree(m_0, [k_0\ k_1\ k_2\ k_3])$

    $c_1 = CipherThree(m_1, [k_0\ k_1\ k_2\ k_3])$

    for $k_3 = 0$ *until* 15 **do**

        $y_0 = S^{-1}(k_3 \oplus c_0)$

        $y_1 = S^{-1}(k_3 \oplus c_1)$

        $\delta_y = y_0 \oplus y_1$

        if $\delta_y == 0x0c$ **then**

            $counter(k_3) = counter(k_3) + 1$

        end

    end

end

$key = argmax(counter)$

if correct $\frac{12}{16} \times \frac{6}{16} \approx 0.279$

if wrong random select

$\frac{1}{16} = 0.0625.$

- Notice that the attack on CipherThree is identical to the attack on CipherTwo with the exception of choosing another differential

CipherFour

CipherFour

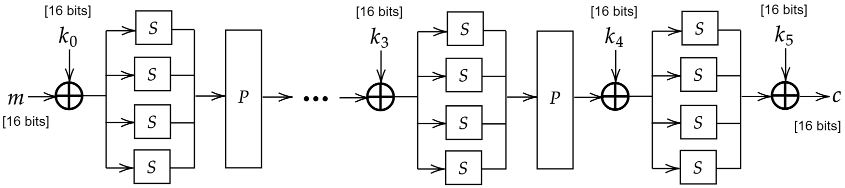
- CipherFour has 5 rounds and plaintext/ciphertext blocklength of 16 bits
- CipherFour uses 6 keys $k_0, k_1, k_2, k_3, k_4, k_5$ of 16 bits each

- **CipherFour encryption algorithm**

```
CipherFour(m, [k_0 k_1 k_2 k_3 k_4 k_5])
u_0 = m
for i=1 to 4 do
    Add the key  a_i = u_{i-1} ⊕ k_{i-1}
    Split a_i to four nibbles  [A_0, A_1, A_2, A_3]
    Apply the sbox  t_i = [S(A_0), S(A_1), S(A_2), S(A_3)]
    Apply the permutation  u_i = P(t_i)
end
Add the key  a_5 = u_4 ⊕ k_4
Split a_5 to four nibbles  [A_0, A_1, A_2, A_3]
Apply the sbox  t_5 = [S(A_0), S(A_1), S(A_2), S(A_3)]
Add the key  c = t_5 ⊕ k_5
```

CipherFour

- CipherFour with 16-bit input m, 6×16-bit roundkeys $(k_0, k_1, k_2, k_3, k_4, k_5)$ and 16-bit ciphertext c



input split into 4×4bit

CipherFour

- We use the previous 4×4 lightweight sbox, that maps input x to $S(x)$

x	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$S(x)$	6	4	c	5	0	7	2	e	1	f	3	d	8	a	9	b

- We use the following bit-level permutation that maps bit position i to $P(i)$

i	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
$P(i)$	0	4	8	12	1	5	9	13	2	6	10	14	3	7	11	15

CipherFour

- To perform differential cryptanalysis on CipherFour we need to find a combination of differential characteristics that predicts the difference δ after the penultimate round (round 4)
- If the combination has high enough probability, we can work backwards from the ciphertext, invert the sbox and recover k_5

CipherFour

- To perform differential cryptanalysis on CipherFour we need to find a combination of differential characteristics that predicts the difference δ after the penultimate round (round 4)
- If the combination has high enough probability, we can work backwards from the ciphertext, invert the sbox and recover k_5

- We start by finding a **single-round differential characteristic** across the key addition, sbox $S(\cdot)$ and permutation $P(\cdot)$ operations

CipherFour

- To perform differential cryptanalysis on CipherFour we need to find a combination of differential characteristics that predicts the difference δ after the penultimate round (round 4)
- If the combination has high enough probability, we can work backwards from the ciphertext, invert the sbox and recover k_5

- We start by finding a **single-round differential characteristic** across the key addition, sbox $S(\cdot)$ and permutation $P(\cdot)$ operations

1. We start the round with:
$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3)$$
where α_j denotes the difference δ in nibble j of the round input

CipherFour

- To perform differential cryptanalysis on CipherFour we need to find a combination of differential characteristics that predicts the difference δ after the penultimate round (round 4)
- If the combination has high enough probability, we can work backwards from the ciphertext, invert the sbox and recover k_5

- We start by finding a **single-round differential characteristic** across the key addition, sbox $S(\cdot)$ and permutation $P(\cdot)$ operations

1. We start the round with:
$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3)$$
where α_j denotes the difference δ in nibble j of the round input

2. Then the key is added, an operation that does not change the difference
$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \xrightarrow{addkey} (\alpha_0, \alpha_1, \alpha_2, \alpha_3)$$

CipherFour

3. Then we apply the sbox $S(\cdot)$ to the four nibbles

$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \xrightarrow{S} (\beta_0, \beta_1, \beta_2, \beta_3)$$

CipherFour

3. Then we apply the sbox $S(\cdot)$ to the four nibbles

$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \xrightarrow{S} (\beta_0, \beta_1, \beta_2, \beta_3)$$

4. Finally we apply the permutation $P(\cdot)$ to the 16-bit sbox output

$$(\beta_0, \beta_1, \beta_2, \beta_3) \xrightarrow{P} (\gamma_0, \gamma_1, \gamma_2, \gamma_3)$$

CipherFour

3. Then we apply the sbox $S(\cdot)$ to the four nibbles

$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \xrightarrow{S} (\beta_0, \beta_1, \beta_2, \beta_3)$$

4. Finally we apply the permutation $P(\cdot)$ to the 16-bit sbox output

$$(\beta_0, \beta_1, \beta_2, \beta_3) \xrightarrow{P} (\gamma_0, \gamma_1, \gamma_2, \gamma_3)$$

The single-round differential characteristic can be summarized as:

$$(\alpha_0, \alpha_1, \alpha_2, \alpha_3) \xrightarrow{\mathcal{R}} (\gamma_0, \gamma_1, \gamma_2, \gamma_3)$$

We will now show which differential characteristics to choose across every operation

CipherFour

- To choose an efficient (attack-wise) characteristic over the sbox we construct the **difference distribution table** for $S(\cdot)$
- Every table entry $(\delta_{in}, \delta_{out})$ gives (once divided by 16) the probability that the difference δ_{in} between sbox inputs yields difference δ_{out} between sbox outputs

δ_{in} \ δ_{out}	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	6	-	-	-	-	2	-	2	-	-	2	-	4	-
2	-	6	6	-	-	-	-	-	-	2	2	-	-	-	-	-
3	-	-	-	6	-	2	-	-	2	-	-	-	4	-	2	-
4	-	-	-	2	-	2	4	-	-	2	2	2	-	-	2	-
5	-	2	2	-	4	-	-	4	2	-	-	2	-	-	-	-
6	-	-	2	-	4	-	-	2	2	-	2	2	2	-	-	-
7	-	-	-	-	-	4	4	-	2	2	2	2	-	-	-	-
8	-	-	-	-	-	2	-	2	4	-	-	4	-	2	-	2
9	-	2	-	-	-	2	2	2	-	4	2	-	-	-	-	2
a	-	-	-	-	2	2	-	-	-	4	4	-	2	2	-	-
b	-	-	-	2	2	-	2	2	2	-	-	4	-	-	2	-
c	-	4	-	2	-	2	-	-	2	-	-	-	-	-	6	-
d	-	-	-	-	-	-	2	2	-	-	-	-	6	2	-	4
e	-	2	-	4	2	-	-	-	-	-	2	-	-	-	-	6
f	-	-	-	-	2	-	2	-	-	-	-	-	-	10	-	2

CipherFour

- A good (but not always useful) choice of differential characteristic is $\delta_{in} = 0$, resulting in $\delta_{out} = 0$

$$\texttt{0x00} \xrightarrow{S} \texttt{0x00}, \quad \text{with probability } 16/16{=}1$$

δ_{in} \ δ_{out}	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	**16**	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	6	-	-	-	-	2	-	2	-	-	2	-	4	-
2	-	6	6	-	-	-	-	-	-	2	2	-	-	-	-	-
3	-	-	-	6	-	2	-	-	2	-	-	-	4	-	2	-
4	-	-	-	2	-	2	4	-	-	2	2	2	-	-	2	-
5	-	2	2	-	4	-	-	4	2	-	-	2	-	-	-	-
6	-	-	2	-	4	-	-	2	2	-	2	2	2	-	-	-
7	-	-	-	-	-	4	4	-	2	2	2	2	-	-	-	-
8	-	-	-	-	-	2	-	2	4	-	-	4	-	2	-	2
9	-	2	-	-	-	2	2	2	-	4	2	-	-	-	-	2
a	-	-	-	-	2	2	-	-	-	4	4	-	2	2	-	-
b	-	-	-	2	2	-	2	2	2	-	-	4	-	-	2	-
c	-	4	-	2	-	2	-	-	2	-	-	-	-	-	6	-
d	-	-	-	-	-	-	2	2	-	-	-	-	6	2	-	4
e	-	2	-	4	2	-	-	-	-	-	2	-	-	-	-	6
f	-	-	-	-	2	-	2	-	-	-	-	-	-	10	-	2

CipherFour

- In CipherTwo and CipherThree we used the best choice at hand

$$\texttt{0x0f} \xrightarrow{S} \texttt{0x0d}, \quad \text{with probability } 10/16$$

- A greedy choice may not always be optimal when we have several sboxes per round and several rounds to combine

δ_{in} \ δ_{out}	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	6	-	-	-	-	2	-	2	-	-	2	-	4	-
2	-	6	6	-	-	-	-	-	-	2	2	-	-	-	-	-
3	-	-	-	6	-	2	-	-	2	-	-	-	4	-	2	-
4	-	-	-	2	-	2	4	-	-	2	2	2	-	-	2	-
5	-	2	2	-	4	-	-	4	2	-	-	2	-	-	-	-
6	-	-	2	-	4	-	-	2	2	-	2	2	2	-	-	-
7	-	-	-	-	-	4	4	-	2	2	2	2	-	-	-	-
8	-	-	-	-	-	2	-	2	4	-	-	4	-	2	-	2
9	-	2	-	-	-	2	2	2	-	4	2	-	-	-	-	2
a	-	-	-	-	2	2	-	-	-	4	4	-	2	2	-	-
b	-	-	-	2	2	-	2	2	2	-	-	4	-	-	2	-
c	-	4	-	2	-	2	-	-	2	-	-	-	-	-	6	-
d	-	-	-	-	-	-	2	2	-	-	-	-	6	2	-	4
e	-	2	-	4	2	-	-	-	-	-	2	-	-	-	-	6
f	-	-	-	-	2	-	2	-	-	-	-	-	-	**10**	-	2

CipherFour

- In CipherFour we will use:

$$\text{in the 3rd nibble: } 0\text{x}02 \xrightarrow{S} 0\text{x}02, \quad \text{with probability } 6/16$$

$$\text{in rest of the nibbles: } 0\text{x}00 \xrightarrow{S} 0\text{x}00, \quad \text{with probability } 1$$

why 3rd nibble?

why only? only 1 changing nibble is enough.

why 3rd? the permutation bit fall into exactly 1 bit per round.

not fully scattered,

like if 3rd is 0x2, only bit 4 → bit 6.

only mess 1 bit pos.

CipherFour

- In CipherFour we will use:

$$\text{in the 3rd nibble: } \texttt{0x02} \xrightarrow{S} \texttt{0x02}, \quad \text{with probability } 6/16$$

$$\text{in rest of the nibbles: } \texttt{0x00} \xrightarrow{S} \texttt{0x00}, \quad \text{with probability } 1$$

- Combining the four nibbles, this is stated as:

$$(\texttt{0x00}, \texttt{0x00}, \texttt{0x02}, \texttt{0x00}) \xrightarrow{S} (\texttt{0x00}, \texttt{0x00}, \texttt{0x02}, \texttt{0x00})$$

$$\text{with probability } 1 * 1 * 6/16 * 1 = 6/16$$

Notice that this particular characteristic over $S(\cdot)$ does not alter the differences

CipherFour

- The permutation $P(\cdot)$ is a linear operation, thus we can obtain a differential characteristic with probability 1

$$(\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}) \xrightarrow{P} (\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}), \quad \text{with probability } 1$$

Notice that this particular characteristic over $P(\cdot)$ does not alter the differences

CipherFour

- The permutation $P(\cdot)$ is a linear operation, thus we can obtain a differential characteristic with probability 1

$$(0x00, 0x00, 0x02, 0x00) \xrightarrow{P} (0x00, 0x00, 0x02, 0x00), \quad \text{with probability } 1$$

  Notice that this particular characteristic over $P(\cdot)$ does not alter the differences
- Thus the one-round differential characteristic is summarized as:

$$(0x00, 0x00, 0x02, 0x00) \xrightarrow{\mathcal{R}} (0x00, 0x00, 0x02, 0x00), \quad \text{with probability } 6/16$$

CipherFour

- The permutation $P(\cdot)$ is a linear operation, thus we can obtain a differential characteristic with probability 1

$$(\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}) \xrightarrow{P} (\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}), \quad \text{with probability } 1$$

  Notice that this particular characteristic over $P(\cdot)$ does not alter the differences
- Thus the one-round differential characteristic is summarized as:

$$(\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}) \xrightarrow{\mathcal{R}} (\mathtt{0x00}, \mathtt{0x00}, \mathtt{0x02}, \mathtt{0x00}), \quad \text{with probability } 6/16$$

- Such a characteristic is called **iterative**, since it can be combined with itself over any number of rounds
- We apply the round characteristic for four rounds of CipherFour, in order to attack the 5th cipher round

$$(0,0,2,0) \xrightarrow{\mathcal{R}} (0,0,2,0) \xrightarrow{\mathcal{R}} (0,0,2,0) \xrightarrow{\mathcal{R}} (0,0,2,0) \xrightarrow{\mathcal{R}} (0,0,2,0)$$

$$\text{with probability } (6/16)^4$$

CipherFour

- We have constructed a 4-round characteristic with probability $(6/16)^4 = 0.0198$
- The probability of a difference occurring at random is $(1/16) = 0.0625$
  i.e. higher than the 4-round characteristic, making our construction less useful

CipherFour

- We have constructed a 4-round characteristic with probability $(6/16)^4 = 0.0198$
- The probability of a difference occurring at random is $(1/16) = 0.0625$
  i.e. higher than the 4-round characteristic, making our construction less useful

- The problem is that many plaintext pairs (all with difference $(0, 0, 2, 0)$) are not following the constructed 4-round characteristic
- We refer to the plaintext pairs that follow the 4-round characteristic as **right pairs** and the ones that do not as **wrong pairs**
- We can often eliminate a wrong plaintext pair by looking into the respective ciphertext pair. The process is called **filtering**.

CipherFour

- **Filtering in CipherFour.** We focus on the difference observed at the 16-bit output of the penultimate round (round 4). If we have a right pair then:

$$\delta_{u_4} = (0, 0, 2, 0)$$

CipherFour

- **Filtering in CipherFour.** We focus on the difference observed at the 16-bit output of the penultimate round (round 4). If we have a right pair then:

$$\delta_{u_4} = (0, 0, 2, 0)$$

- The key addition during the 5th round does not affect the difference δ_{a_5}

$$(0, 0, 2, 0) \xrightarrow{addkey} (0, 0, 2, 0)$$

CipherFour

- **Filtering in CipherFour.** We focus on the difference observed at the 16-bit output of the penultimate round (round 4). If we have a right pair then:

$$\delta_{u_4} = (0, 0, 2, 0)$$

- The key addition during the 5th round does not affect the difference δ_{a_5}

$$(0, 0, 2, 0) \xrightarrow{addkey} (0, 0, 2, 0)$$

- The sbox during the 5th round sbox does affect the difference δ_{t_5}.
  In particular, $0 \xrightarrow{S} 0$ but $2 \xrightarrow{S} h,$ where $h \in \{1, 2, 9, \mathtt{a}\}$

Combining the four nibbles: $(0, 0, 2, 0) \xrightarrow{S} (0, 0, h, 0)$

where $h \in \{1, 2, 9, \mathtt{a}\}$

CipherFour

δ_{in} \ δ_{out}	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	16	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
1	-	-	6	-	-	-	-	2	-	2	-	-	2	-	4	-
2	-	**6**	**6**	-	-	-	-	-	-	**2**	**2**	-	-	-	-	-
3	-	-	-	6	-	2	-	-	2	-	-	-	4	-	2	-
4	-	-	-	2	-	2	4	-	-	2	2	2	-	-	2	-
5	-	2	2	-	4	-	-	4	2	-	-	2	-	-	-	-
6	-	-	2	-	4	-	-	2	2	-	2	2	2	-	-	-
7	-	-	-	-	-	4	4	-	2	2	2	2	-	-	-	-
8	-	-	-	-	-	2	-	2	4	-	-	4	-	2	-	2
9	-	2	-	-	-	2	2	2	-	4	2	-	-	-	-	2
a	-	-	-	-	2	2	-	-	-	4	4	-	2	2	-	-
b	-	-	-	2	2	-	2	2	2	-	-	4	-	-	2	-
c	-	4	-	2	-	2	-	-	2	-	-	-	-	-	6	-
d	-	-	-	-	-	-	2	2	-	-	-	-	6	2	-	4
e	-	2	-	4	2	-	-	-	-	-	2	-	-	-	-	6
f	-	-	-	-	2	-	2	-	-	-	-	-	-	10	-	2

- Since the sbox is the last CipherFour operation, all 4 possible differences $\{1, 2, 9, a\}$ can appear as ciphertext difference h

CipherFour

- **CipherFour filtering algorithm**

Filter(c_0, c_1)
$\delta_c = c_0 \oplus c_1$
$check_1 = \ \delta_c == (0, 0, 1, 0)$
$check_2 = \ \delta_c == (0, 0, 2, 0)$
$check_3 = \ \delta_c == (0, 0, 9, 0)$
$check_4 = \ \delta_c == (0, 0, \mathrm{a}, 0)$
if *check*$_1$ *or* *check*$_2$ *or* *check*$_3$ *or* *check*$_4$ **then**
    Store ciphertext pair (c_0, c_1)
end

- All the stored ciphertext pairs will be used during the differential cryptanalysis attack, since they originate from difference $(0, 0, 2, 0)$ in the output of the penultimate round (round 4)

CipherFour

```
Generate n random 16-bit plaintext pairs (m_0, m_1) with fixed difference (0,0,2,0)
Compute n respective ciphertext pairs (c_0, c_1)
Apply filtering and keep m out of n ciphertext pairs (c_0, c_1)
counter(0 until 15) = [0, 0, ..., 0]
for i = 1 until m do
    for k_6 = 0 until 15 do
        q_0 = S^{-1}(k_6 ⊕ c_0)
        q_1 = S^{-1}(k_6 ⊕ c_1)
        δ_q = q_0 ⊕ q_1
        if δ_q == 0x02 then
            counter(k_6) = counter(k_6) + 1
        end
    end
end
key = argmax(counter)
```

CipherFour

Final notes on Differential Cryptanalysis:

- Strong attack that has been applied to many ciphers
- It requires finding differential characteristics across many cipher rounds
- It has many extensions: impossible differentials, higher-order differentials, truncated differentials