TU Graz

SCIENCE
PASSION
TECHNOLOGY

Symmetric Primitives: Design & Cryptanalysis

Lorenzo Grassi

November 2019

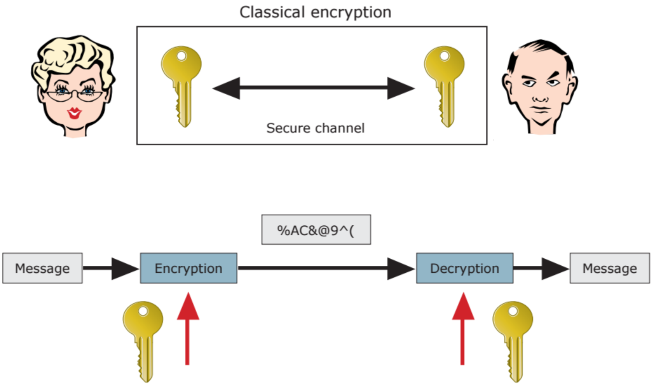
Outline

- Symmetric Cryptography
  - SPN Design
- Differential Cryptanalysis
  - Toy Ciphers
- Secure designs against differential cryptanalysis
  - Wide Trail Design Strategy: AES
- Closing Remarks

Symmetric Cryptography and SPN Design

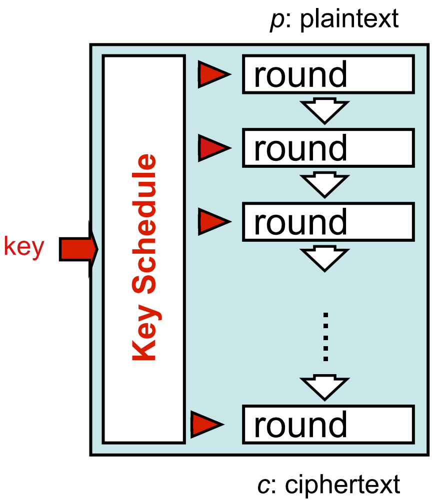
Symmetric Cryptography

Cryptography is communication in the presence of an adversary (Ron Rivest)



Reprinted from https://www.cosic.esat.kuleuven.be/summer_school_sardinia_2015/slides/LRKnudsen.pdf by *Lars R. Knudsen*

Design Principles – SPN Construction



Design Principles – SPN Construction

- Iterative ciphers:

$$AddKey \circ (M_r \circ S_r) \circ AddKey \circ \ldots \circ (M_2 \circ S_2) \circ AddKey \circ (M_1 \circ S_1) \circ AddKey(\cdot)$$

$\rightarrow$ Round transformation

$$round(\cdot) \equiv AddKey \circ M_i \circ S_i(\cdot)$$

- Often:
  - $M_i \circ S_i = M \circ S$
  - $AddKey_K(\cdot) = \cdot + K$ (over $(\mathbb{F}, +, \times)$)

Notation: $M \equiv$ Mixing Layer - $S \equiv$ S-Box Layer

Practical Constraints

Problem: Hardware/Software Constraints!

Typically

- *Small* substitution elements
- Key Agility:
  - the amount of preprocessing of a key (e.g. key-schedule) should be small
- the state size of a cipher uses is small (e.g. 16/32 bytes)

Design Principles – S-Box

- An S-Box substitutes a small block of bits (the input of the S-Box) by another block of bits (the output of the S-Box)

- An S-Box is usually *not* simply a permutation of the bits: it must provide non-linearity!

  If no operation provides non-linearity, each ciphertext c can be decomposed as

  $$c = L_0(p) \oplus L_1(k) \qquad L_0(\cdot), L_1(\cdot) \text{ linear}$$

  and the key can be easily found (if plaintext p is known).

- The S-Box must be invertible in SPN constructions, while it is not required for Feistel Constructions!

Design Principles – S-Box

- An S-Box substitutes a small block of bits (the input of the S-Box) by another block of bits (the output of the S-Box)

- An S-Box is usually *not* simply a permutation of the bits: it must provide non-linearity!

  If no operation provides non-linearity, each ciphertext c can be decomposed as

  $$c = L_0(p) \oplus L_1(k) \qquad L_0(\cdot), L_1(\cdot) \text{ linear}$$

  and the key can be easily found (if plaintext p is known).

- The S-Box must be invertible in SPN constructions, while it is not required for Feistel Constructions!

Design Principles – S-Box

- An S-Box substitutes a small block of bits (the input of the S-Box) by another block of bits (the output of the S-Box)

- An S-Box is usually *not* simply a permutation of the bits: it must provide non-linearity!

  If no operation provides non-linearity, each ciphertext c can be decomposed as

  $$c = L_0(p) \oplus L_1(k) \qquad L_0(\cdot), L_1(\cdot) \text{ linear}$$

  and the key can be easily found (if plaintext p is known).

- The S-Box must be invertible in SPN constructions, while it is not required for Feistel Constructions!

Design Principles - Mixing

- Linear permutation of all the bits: it takes the outputs of several - if possible, all - S-Boxes of one round, permutes the bits, and feeds them into the S-Boxes of the next round

- A good linear permutation has the property that the output bits of any S-Box are distributed to as many S-Box inputs as possible

- "Branch Number" is a parameter that measures how good a linear permutation is

  (discuss in details in the following)

Design Principles - Mixing

- Linear permutation of all the bits: it takes the outputs of several - if possible, all - S-Boxes of one round, permutes the bits, and feeds them into the S-Boxes of the next round

- A good linear permutation has the property that the output bits of any S-Box are distributed to as many S-Box inputs as possible

- “Branch Number” is a parameter that measures how good a linear permutation is

  (discuss in details in the following)

Should we worry?

- For almost all choices of a round transformation,

$$E_k(\cdot) = \lim_{r \to \infty} (M_i \circ S_i)^r(\cdot)$$

gives a secure (but not practical...) cipher

- Goal: *find the optimal number of rounds to guarantee security and good performances!*

Differential Cryptanalysis

Secure Ciphers - Symmetric Encryption

How can you tell if a cipher is secure?

Definition (Kerckhoffs' Principle)

The security of a cryptosystem must lie in the choice of its keys only. *Everything else (including the algorithm itself) should be considered public knowledge*.

A cipher is secure if there is no attack better than brute force: a solid cipher must resist *all* known attacks!

Secure Ciphers - Symmetric Encryption

How can you tell if a cipher is secure?

Definition (Kerckhoffs' Principle)

The security of a cryptosystem must lie in the choice of its keys only. *Everything else (including the algorithm itself) should be considered public knowledge*.

A cipher is secure if there is no attack better than brute force: a solid cipher must resist all known attacks!

Why Not Brute Force Attack? (1/2)

Given a plaintext and the corresponding ciphertext, *brute-force attack* involves systematically checking all possible key combinations until the correct key is found.

Key Size	*Possible Combinations*
1 bit	2
2 bit	4
4 bit	16
8 bit	256
16 bit	65536
32 bit	$4.2 \cdot 10^{9}$
56 bit (DES)	$7.2 \cdot 10^{16}$
64 bit	$1.8 \cdot 10^{19}$
128 bit (AES-128)	$3.4 \cdot 10^{38}$
192 bit (AES-192)	$6.2 \cdot 10^{57}$
256 bit (AES-256)	$1.1 \cdot 10^{77}$

Why Not Brute Force Attack? (2/2)

(Currently) Fastest Supercomputer ("Summit" - USA, June 2018): $\approx$ 122.3 PetaFLOPS
[FLOPS = Floating point operations per second]

Assume only 1 "floating point operation" is required per combination check (very optimistic!)

Number of combination checks per second: $122.3 \cdot 10^{15}$

Key Size	*Time to Crack*
56 bit (DES)	< 1 sec
128 bit (AES-128)	$8.8 \cdot 10^{13}$ years
192 bit (AES-192)	$1.6 \cdot 10^{33}$ years
256 bit (AES-256)	$3.0 \cdot 10^{52}$ years

For comparison, our universe was born about $1.37 \cdot 10^{10}$ years ago.

Cryptanalysis

Question: *How can we be sure an attacker will require a large amount of work to break a non-perfect system with every method?*

Hard to achieve! But we can at least:

- *make it secure against all known attacks (Symmetric Crypto)*
- reduce it to some “known difficult” problem (Asymmetric Crypto)

Symmetric Cryptanalysis

- Meet-In-The-Middle
- *Differential Cryptanalysis*:
  - Truncated Diff. Cryptanalysis, Impossible Diff. Cryptanalysis, Boomerang Attack, Yoyo Attack, ...
- Linear Cryptanalysis
- Algebraic Attack
  - Interpolation Attack, Higher-Order Differential Attack, Cube Attack, Gröbner Basis Attack, GCD Attack, ...
- Integral/Square Attack
- Related-Key Attack
- ...

Differential Cryptanalysis: The Idea

Idea: *Deduce information about the secret key by tracing differences between pairs of plaintexts during the encryption (and decryption)*

Differential Cryptanalysis is based on the fact that pairs of plaintexts with certain differences yield other differences in the corresponding ciphertexts with a non-uniformity probability distribution.

- First published by Biham and Shamir to attack DES (1993)
- One of the best attack methods in cryptanalysis and applicable to many iterated block ciphers
- Chosen-Plaintext attack

Differential Cryptanalysis: The Idea

Idea: *Deduce information about the secret key by tracing differences between pairs of plaintexts during the encryption (and decryption)*

Differential Cryptanalysis is based on the fact that pairs of plaintexts with certain differences yield other differences in the corresponding ciphertexts with a non-uniformity probability distribution.

- First published by Biham and Shamir to attack DES (1993)
- One of the best attack methods in cryptanalysis and applicable to many iterated block ciphers
- Chosen-Plaintext attack

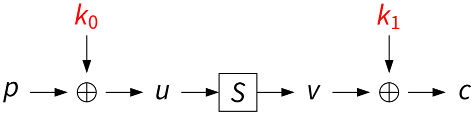
A Simple Block Cipher

The block cipher $E_{k_0||k_1}(p)$ encrypts 4 bits of plaintext using two 4-bit keys:

$$c = E_{k_0||k_1}(p) = S(p \oplus k_0) \oplus k_1$$

The 4-bit S-Box S is defined as follows:

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

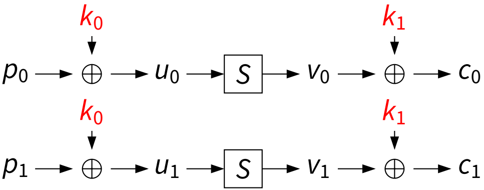


Given $(p_0, c_0) = (\mathtt{a}, 9)$ and $(p_1, c_1) = (5, 6)$, determine the key!?

Brute force (exhaustive search): try all $2^4 \cdot 2^4 = 256$ keys.

The Basic Idea

Assume we know two plaintext/cipher text pairs $(p_0, c_0), (p_1, c_1)$:



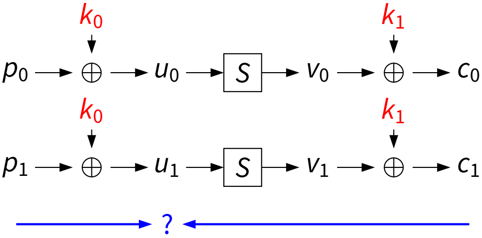
Observation

We know that

$$
\begin{aligned}
u_0 \oplus u_1 &= (p_0 \oplus k_0) \oplus (p_1 \oplus k_0) = p_0 \oplus p_1 \\
v_0 \oplus v_1 &= (c_0 \oplus k_1) \oplus (c_1 \oplus k_1) = c_0 \oplus c_1,
\end{aligned}
$$

even though we do not know k_0 and k_1.

Differential Attack

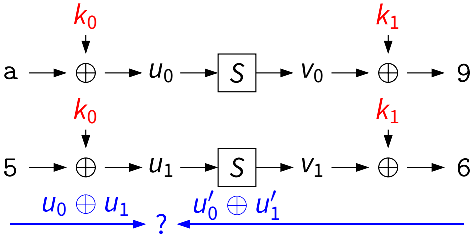


Strategy:

1. compute $u_0 \oplus u_1$
2. guess k_1 (iterate over all values)
3. compute $u_0' = S^{-1}(c_0 \oplus k_1')$ and $u_1' = S^{-1}(c_1 \oplus k_1')$
4. check if $u_0 \oplus u_1 = u_0' \oplus u_1'$
5. if not: key guess was definitely wrong! (filtering)

Example

Given $(p_0, c_0) = (\mathtt{a}, 9)$ and $(p_1, c_1) = (5, 6)$



1. compute $u_0 \oplus u_1 = p_0 \oplus p_1 = \mathtt{a} \oplus 5 = \mathtt{f}$
2. guess k_1 and compute $u'_0 \oplus u'_1$:

k'_1	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
$u'_0 \oplus u'_1$	e	b	e	e	d	8	d	f	f	d	8	d	e	e	b	e

3. only two candidates for k_1 are valid: $k_1 \in \{7, 8\}$

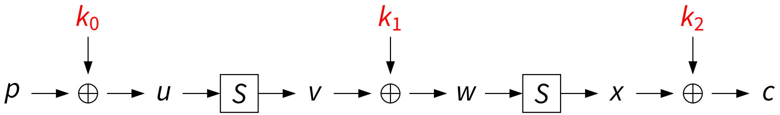
Let's Extend it to a Two-Round Cipher

The block cipher $E_{k_0||k_1||k_2}(p)$ encrypts 4 bits of plaintext using three 4-bit keys:

$$c = E_{k_0||k_1||k_2}(p) = S(S(p \oplus k_0) \oplus k_1) \oplus k_2$$

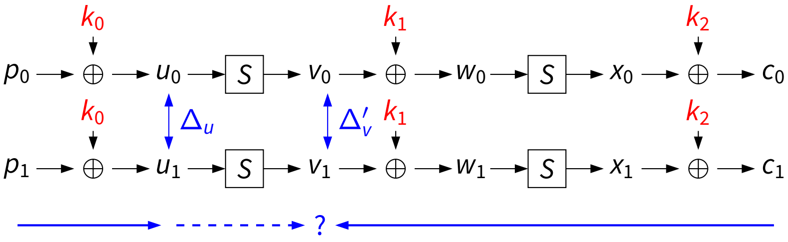
We use the same 4-bit S-Box S:

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



Brute force: $2^{4+4+4} = 4096$ keys.

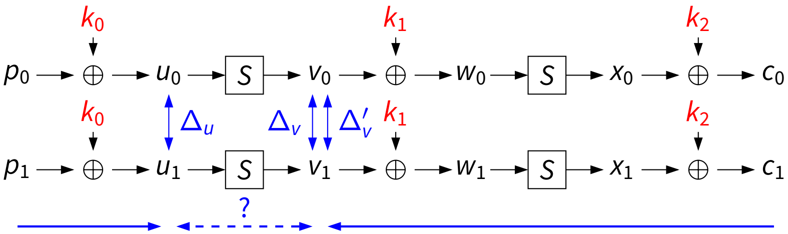
Differential Attack on a Two-Round Cipher



What can we compute?

- compute $\Delta u = \Delta p = p_0 \oplus p_1$
- we guess the last round key k_2 and compute x'_0, x'_1, w'_0, w'_1
- compute $\Delta v' = \Delta w' = w'_0 \oplus w'_1$
- how can we check if our guess for k_2 was correct?

Differential Attack on a Two-Round Cipher



- we want to verify our computed $\Delta v'$ (based on guess k'_2)
- can we get more information about the real Δv?
- using $\Delta u = \texttt{f}$ but not knowing u_0 or u_1?

$\Rightarrow$ let's compute all possible differences for Δv

The Influence of the S-Box (1/2)

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

Only 4 differences for Δv are possible (for the given $\Delta u = \mathtt{f}$).

One difference ($\Delta v = \mathtt{d}$) occurs very often.

The Influence of the S-Box (2/2)

Observations

- the differences are unevenly distributed
- the difference d occurs 10 out of 16 times
- not all differences occur

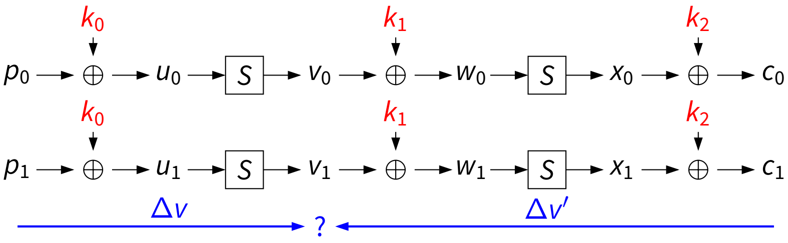
$\Rightarrow$ let's assume that $\Delta v = \mathtt{d}$

Then, we can verify our guess k_2' by checking whether $\Delta v' (= \Delta v) = \mathtt{d}$

With a probability of 10/16 our assumption is right

$\Delta v = v_0 \oplus v_1$
d
d
6
d
d
4
d
f
f
d
4
d
d
6
d
d

Differential Attack on a Two-Round Cipher



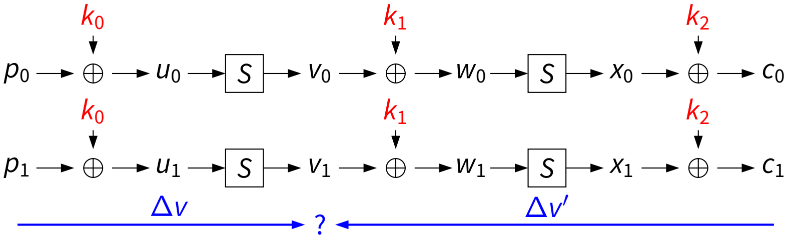
We filter wrong key guesses:

Strategy (1/2)

1. consider 16 plaintext/ciphertext pairs (p_0^i, c_0^i) and (p_1^i, c_1^i) *such that* $p_0^i \oplus p_1^i = \mathtt{f}$ for $i = 0, \ldots, 15$

2. guess the last round key k_2 (iterate over all values)

Differential Attack on a Two-Round Cipher



Strategy (2/2)

3. for each plaintexts-ciphertexts pair: compute x_0', x_1', w_0', w_1' and count the number of pairs for which $\Delta v' = \Delta w' = \mathtt{d}$;

4. we expect that

   - for the *right key*, approx. $16 \cdot \frac{10}{16} = 10$ pairs satisfy $\Delta v' = \mathtt{d}$;
   - for a *wrong key*, approx. $16 \cdot \frac{1}{16} = 1$ pair satisfies $\Delta v' = \mathtt{d}$.

Difference Distribution Table

How can we find differences with a good probability?

- compute all possible output differences for all input differences of an S-Box
- or equivalently: compute the number of solutions x to the equation

$$S(x \oplus \Delta u) \oplus S(x) = \Delta v$$

Definition

Let f be an n to m bit function. The difference distribution table of f is an $2^n \times 2^m$ table whose entries are the number of valid solutions x for each differential $\Delta u \to \Delta v$.

Difference Distribution Table

$\Delta_{in} \setminus \Delta_{out}$	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

Maximum Differential Probability

Let f defined as before. *Differential Probability* is defined as

$$DP^f(\Delta u \to \Delta v) := Pr[f(x \oplus \Delta u) \oplus f(x) = \Delta v] = \\ = \frac{|\{x \text{ s.t. } f(x \oplus \Delta u) \oplus f(x) = \Delta v\}|}{2^n}$$

Definition

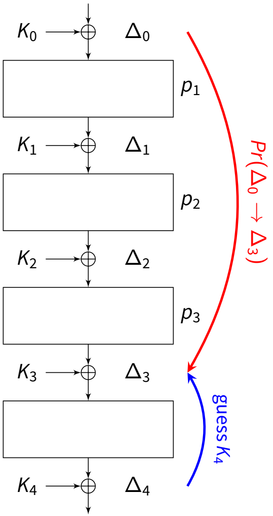
Maximum Differential Probability is defined as

$$DP^f_{max} := \max_{\Delta u \neq 0, \Delta v} DP^f(\Delta u \to \Delta v).$$

In the previous example

$$DP^S_{max} = \max_{\Delta u \neq 0, \Delta v} \frac{|\{x \text{ s.t. } S(x \oplus \Delta u) \oplus S(x) = \Delta v\}|}{16} = \frac{10}{16}$$

Basic Approach of a Differential Attack



1. Find "good" differential characteristic

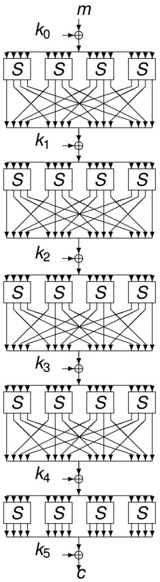
$$\Delta_0 \to \Delta_1 \to \Delta_2 \to \Delta_3$$

2. Guess final key K_4' and compute backward through the S-Boxes to determine Δ_3'

3. The right key satisfies $\Delta_3' = \Delta_3$ with probability $Pr(\Delta_0 \to \Delta_3)$, while a wrong key satisfies $\Delta_3' = \Delta_3$ with probability $1/|\mathcal{P}| = 2^{-n}$ ($\mathcal{P} = \mathbb{F}_2^n$ is the plaintext space).

4. *Necessary condition* for the attack: $Pr(\Delta_0 \to \Delta_3) \gg 1/|\mathcal{P}| = 2^{-n}$.

A 4-round Toy-Cipher



The 4-bit S-Box S - remember: $y = S(x)$ - is defined as

x	0	1	2	3	4	5	6	7
$S(x)$	6	4	c	5	0	7	2	e

x	8	9	a	b	c	d	e	f
$S(x)$	1	f	3	d	8	a	9	b

Bit (linear) permutation P - remember: $y_{P(i)} = x_i$ - is defined as

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

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

1-round Characteristic

The 1-round characteristic for ToyCipher

$$(0, 0, 2, 0) \to (0, 0, 2, 0)$$

holds with probability $6/16$ due to the following facts

1. equation $S(x \oplus 2) \oplus S(x) = 2$ admits 6 different solutions:

$\to$ probability that input difference 2 is mapped to output difference 2 is $6/16$

2. details of the permutation layer

Characteristic and Differential

The 1-round characteristic for ToyCipher

$$(0,0,2,0) \to (0,0,2,0)$$

holds with probability $6/16$.

The 4-round characteristic for ToyCipher

$$(0,0,2,0) \to (0,0,2,0) \to (0,0,2,0) \to (0,0,2,0) \to (0,0,2,0)$$

holds with probability $(6/16)^4 = \frac{81}{4096}$.

The 4-round differential for ToyCipher

$$(0,0,2,0) \to ? \to ? \to ? \to (0,0,2,0)$$

holds with probability higher than $\frac{324}{4096}$.

Characteristic and Differential

The 1-round characteristic for ToyCipher

$$(0,0,2,0) \to (0,0,2,0)$$

holds with probability $6/16$.

The 4-round characteristic for ToyCipher

$$(0,0,2,0) \to (0,0,2,0) \to (0,0,2,0) \to (0,0,2,0) \to (0,0,2,0)$$

holds with probability $(6/16)^4 = \frac{81}{4096}$.

The 4-round differential for ToyCipher

$$(0,0,2,0) \to ? \to ? \to ? \to (0,0,2,0)$$

holds with probability higher than $\frac{324}{4096}$.

Characteristic and Differential

- An s-round characteristic is a sequence of differences, denoted as an $(s+1)$-tuple $(\Delta_0, \Delta_1, ..., \Delta_s)$, where Δ_i is the difference between the values of the partially encrypted messages after i rounds and Δ_0 is the difference between the plaintexts.

- An s-round differential is a pair of differences (Δ_0, Δ_s), where Δ_s is the expected difference after s rounds and Δ_0 represents the chosen value of Δm.

Note:

$$Pr(\Delta_0 \to \Delta_s) \geq Pr(\Delta_0 \to \Delta_1 \to ... \to \Delta_s)$$

Wide Trail Design Strategy: AES

Defending against Differential Cryptanalysis

- To prevent Differential Cryptanalysis: *keep the probability of each characteristic/differential as low as possible*

- Since it is difficult to compute the exact probability, compute “bounds”

- Strategy:
  - compute “Maximum Differential Probability” of the S-Box DP_{max}^{SBox}
  - compute a (lower) bound of the number of active S-Boxes

$\Rightarrow$ design S-Boxes with low maximum values DP_{max}

$\Rightarrow$ *design mixing layers which result in many active S-Boxes*

Defending against Differential Cryptanalysis

- To prevent Differential Cryptanalysis: *keep the probability of each characteristic/differential as low as possible*

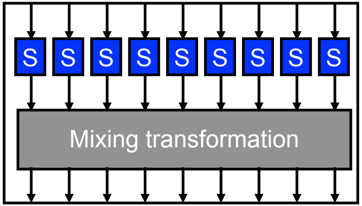
- Since it is difficult to compute the exact probability, compute “bounds”

- Strategy:
  - compute “Maximum Differential Probability” of the S-Box DP^{SBox}_{max}
  - compute a (lower) bound of the number of active S-Boxes

$\Rightarrow$ design S-Boxes with low maximum values DP_{max}

$\Rightarrow$ *design mixing layers which result in many active S-Boxes*

SPN: Single-Round Optimization



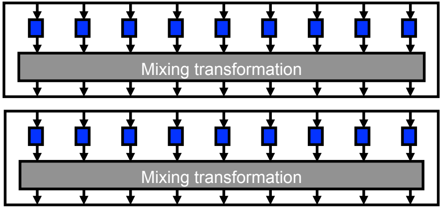
Relevant:

- Number of active components (S-Boxes) in input
- Worst-case (max) differential probability in S-Box

Result:

- bound of 1 active S-Box per round (= minimum number of active S-Boxes) - Mixing Transformation is *irrelevant* (in this case)
- we need to design large S-Boxes with small DP_{max}

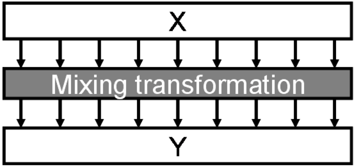
SPN: Two-Round Optimization



- Relevant: number of active S-Boxes in input and after first round
- The number of active S-Boxes after the first round depends on the (linear) *mixing transformation*
  - Branch number $\mathcal{B}$: *minimum number of active S-Boxes in two consecutive rounds*

$\Rightarrow$ Provides a bound of $\mathcal{B}$ active S-Boxes per two rounds

SPN: Designing the Mixing Transformation



Given $Y = m(X)$ linear, then

$$\mathcal{B} \leq 1 + \text{ total number of components (= number of S-Boxes) in } Y$$

$\Rightarrow$ *Design a Mixing Transformation m that maximizes* $\mathcal{B}$

A linear transformation that maximes $\mathcal{B}$ *is called MDS (= Maximum Distance Separable)*

AES: Iterated Block Cipher

Advanced Encryption Standard (128-, 192-, 256-bit keys)

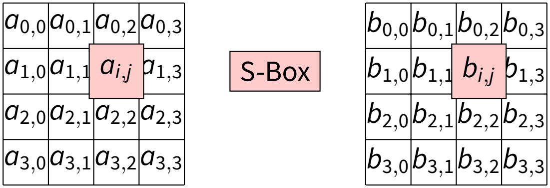
- 10, 12, 14 times applying the same round function
- State of 4×4 bytes (128 bits)

- Round function: composed of 4 steps

$$R_K(\cdot) = K \oplus MC \circ SR \circ \text{S-Box}(\cdot)$$

- Each step has its own particular functionality!

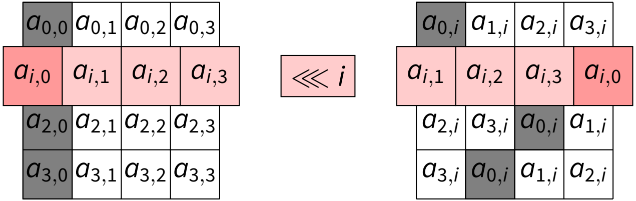
SubBytes



- Bytes are transformed by invertible S-Box with $b_{i,j} = S(a_{i,j})$
- Same S-Box (lookup table) for the whole cipher:
  - based on multiplicative inverse in $GF(2^8)$
  - What about DP_{max}?

$$DP_{max}(\text{AES S-Box}) = \frac{4}{256}$$

ShiftRows



- Rows are rotated over 4 different offsets
- "*Optimal Diffusion*": two bytes in the same column are mapped into different columns after ShiftRows operation

MixColumns

$$\begin{array}{|c|c|c|c|} \hline a_{0,0} & a_{0,1} & a_{0,j} & a_{0,3} \\ \hline a_{1,0} & a_{1,1} & a_{1,j} & a_{1,3} \\ \hline a_{2,0} & a_{2,1} & a_{2,j} & a_{2,3} \\ \hline a_{3,0} & a_{3,1} & a_{3,j} & a_{3,3} \\ \hline \end{array} \quad \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad \begin{array}{|c|c|c|c|} \hline b_{0,0} & b_{0,1} & b_{0,j} & b_{0,3} \\ \hline b_{1,0} & b_{1,1} & b_{1,j} & b_{1,3} \\ \hline b_{2,0} & b_{2,1} & b_{2,j} & b_{2,3} \\ \hline b_{3,0} & b_{3,1} & b_{3,j} & b_{3,3} \\ \hline \end{array}$$

- Columns transformed by 4×4 matrix over $GF(2^8)$
- Linear function with Branch number $\mathcal{B} = 5$
  - MDS matrix (i.e. linear function that maximizes $\mathcal{B}$)
- Together with ShiftRows, *high diffusion* over multiple rounds:
  - minimum $\mathcal{B}^2 = 25$ active S-Boxes for 4 rounds

MixColumns

$$\begin{array}{|c|c|c|c|}\hline a_{0,0} & a_{0,1} & a_{0,j} & a_{0,3} \\ \hline a_{1,0} & a_{1,1} & a_{1,j} & a_{1,3} \\ \hline a_{2,0} & a_{2,1} & a_{2,j} & a_{2,3} \\ \hline a_{3,0} & a_{3,1} & a_{3,j} & a_{3,3} \\ \hline \end{array} \quad \times \begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix} \quad \begin{array}{|c|c|c|c|}\hline b_{0,0} & b_{0,1} & b_{0,j} & b_{0,3} \\ \hline b_{1,0} & b_{1,1} & b_{1,j} & b_{1,3} \\ \hline b_{2,0} & b_{2,1} & b_{2,j} & b_{2,3} \\ \hline b_{3,0} & b_{3,1} & b_{3,j} & b_{3,3} \\ \hline \end{array}$$

- Columns transformed by 4×4 matrix over $GF(2^8)$
- Linear function with Branch number $\mathcal{B} = 5$
  - MDS matrix (i.e. linear function that maximizes $\mathcal{B}$)
- Together with ShiftRows, *high diffusion* over multiple rounds:
  - minimum $\mathcal{B}^2 = 25$ active S-Boxes for 4 rounds

Bounds in AES (1/2)

Diffusion in AES:

- MixColumns: branch number 5
- ShiftRows: Diffusion-optimal

$\Rightarrow$ *lower bound* for number of active S-Boxes per 4 rounds: 25

As example

where: *white byte* = zero XOR-difference - *black byte* = non-zero XOR-difference

Bounds in AES (2/2)

- Diffusion in AES: *lower bound* for number of active S-Boxes per 4 rounds $= 25$
- AES S-Box:
  - differential probability (DP) $\leq 4/256 = 2^{-6}$, that is $DP_{max} = 2^{-6}$
- Provable bound:
  - probability of 4-round AES characteristic $\leq (2^{-6})^{25} = 2^{-150}$
  - remember: given a fixed input difference, each output difference has prob. 2^{-128}

Remark on resistance against DC

- **Remember:** characteristics are not differentials (differentials have much higher probability than characteristics)

- What is the EDP (= Expected Differential Probability) of r-round AES?
  - easy for $r = 1$, i.e. for a single S-Box (DP_{max})
  - for $r = 2$ rounds, EDP has been be computed (exhaustive search)
  - more rounds $r \geq 3$: unknown

- Huge margin anyway (max. 2^{-150} for 4-round characteristic and AES is composed of 10 rounds)

$\Rightarrow$ Standard differential attack on full AES is infeasible

Closing Remarks

- Block ciphers are important in practice
- Design approaches are bottom-up
  - Start from simple components
  - Make something you can’t break
  - Determined by state of the art in cryptanalysis
- Challenges
  - Provable security
  - Performance/cost in constrained platforms

Thanks for your attention!

Questions?

Comments?

Why Prob. 1/16 for a Wrong Guessed Key?

Observe:

$$\begin{aligned}\Delta v' \equiv \Delta w' &= S^{-1}(c_0 \oplus k_2') \oplus S^{-1}(c_1 \oplus k_2') = \\ &= S^{-1}(S(w_0) \oplus k_2 \oplus k_2') \oplus S^{-1}(S(w_1) \oplus k_2 \oplus k_2')\end{aligned}$$

- If $k_2 = k_2'$, then $\Delta v' = w_0 \oplus w_1 = \Delta v$, that is $\Delta v' = \mathtt{d}$ with prob. $10/16$;

- Otherwise, *since k_2 is unknown and uniformly distributed*, it is possible to show that $\Delta v' = \mathtt{d}$ with *approximately* prob. $1/16$

  ("*Wrong-key randomization hypothesis*")