

Block Ciphers

Cryptology (1)

Martin Stanek

2025

KI FMFI UK Bratislava

Introduction

- encryption/decryption

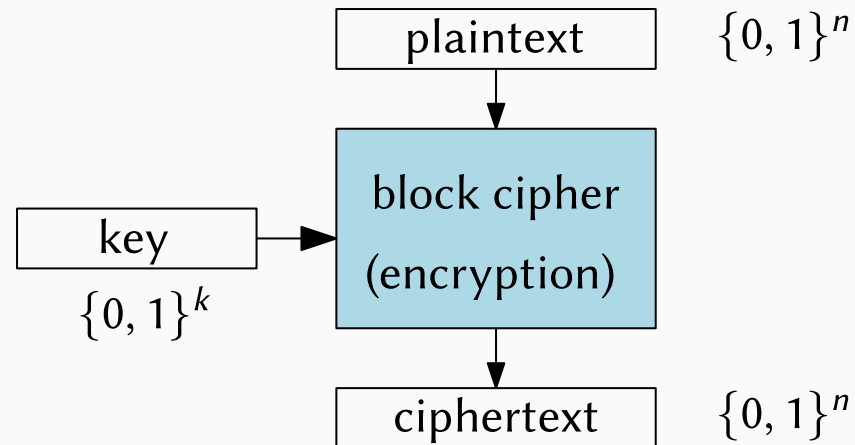
$$E, D : \{0, 1\}^k \times \{0, 1\}^n \rightarrow \{0, 1\}^n$$

- k – key length, n – block length

- correctness:

$$\forall K \in \{0, 1\}^k \quad \forall m \in \{0, 1\}^n : D_K(E_K(m)) = m$$

- E_K and D_K are mutually inverse permutations on $\{0, 1\}^n$



- Examples of real-world block ciphers
 - AES – block length: 128, key lengths: 128, 192, 256
 - TDEA (also known as 3DES) – block length: 64, key lengths: 112, 168
- NIST SP 800-131A rev. 3 (draft, 2024):
 - AES acceptable
 - TDEA encryption disallowed; decryption for legacy use

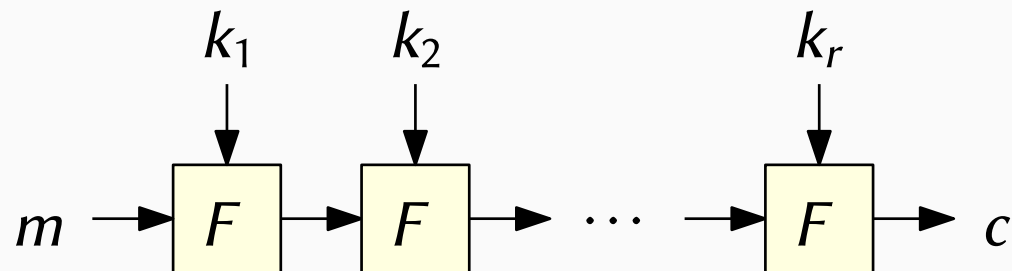
Block size impact on security

- block cipher as a substitution with huge alphabet 2^n
 - frequency analysis impossible
- short block size – (possibly) easier cryptanalysis
- extremely short block size
 - small alphabet (code book can be learned in some attack scenarios)
 - max. $(2^n)!$ permutations, regardless of key length

Key size impact on security

- exhaustive key search (EKS) complexity $\approx 2^k$
 - key length should be sufficiently large
- important assumption: keys with uniform distribution
 - otherwise enumerate keys by their probabilities (in descending order)
 - keys often derived from user passwords (\Rightarrow non-uniformity)
- almost anything with better complexity than EKS is a successful cryptanalytic attack (at least in theory)
 - can still be impractical, because of
 - complexity, e.g. 2^{120} instead of 2^{128} is still infeasible
 - assumptions, e.g. CPA with 2^{90} of chosen plaintext blocks encrypted with the same key is rather unrealistic

Iterated ciphers



- the most frequently used construction method for block ciphers
- iteration of round function $F : \{0, 1\}^{k'} \times \{0, 1\}^n \rightarrow \{0, 1\}^n$
- structure:
 - key scheduling/expansion: producing round keys k_1, \dots, k_r from the key
 - sequential iteration of F (r rounds): $c = F_{k_r}(\dots F_{k_2}(F_{k_1}(m)) \dots)$
 - usually with some form of key whitening: $c = k_{r+1} \oplus F_{k_r}(\dots F_{k_1}(m \oplus k_0) \dots)$
 - sometimes the first/the last round is different
- decryption: inverse round function, reverse order of round keys

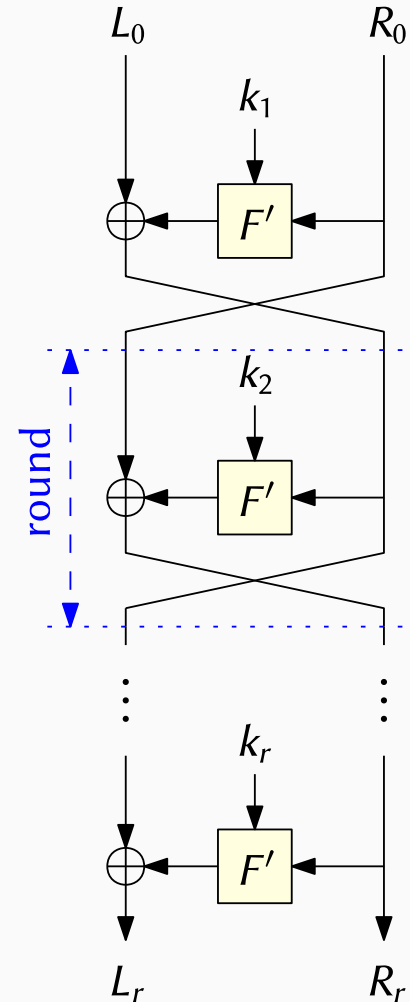
Feistel ciphers

- method of constructing a round function
 - its inverse has the same structure
- decryption \approx encryption (with reversed order of round keys)
 \Rightarrow equal speed of encryption and decryption with precomputed round keys
- plaintext divided into left and right halves: L_0, R_0
- iterations (for $i = 1, \dots, r - 1$):

$$L_i = R_{i-1}, \quad R_i = L_{i-1} \oplus F'_{k_i}(R_{i-1})$$

- last round:

$$L_r = L_{r-1} \oplus F'_{k_r}(R_{r-1}), \quad R_r = R_{r-1}$$

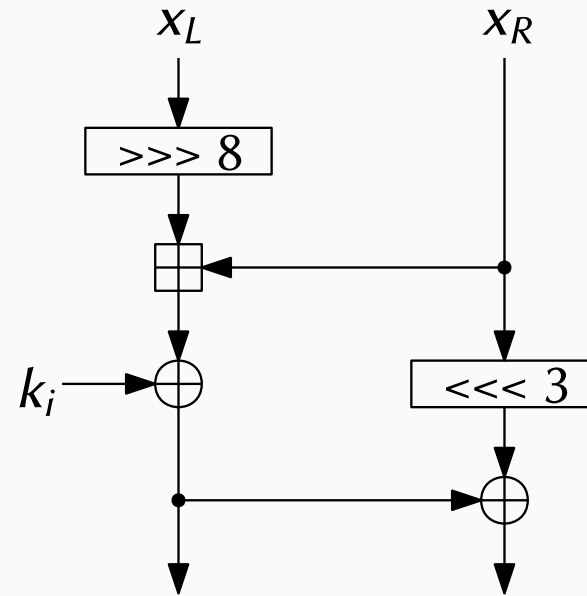


- example: DES (TDEA/3DES)
- generalization: unbalanced Feistel (splitting block into parts of unequal length)
- Feistel network is used in other cryptographic constructions, for example:
 - OAEP (Optimal Asymmetric Encryption Padding) for RSA encryption
 - format preserving encryption
- theoretical construction: pseudorandom function \rightarrow pseudorandom permutation

Speck (an example of a lightweight block cipher)

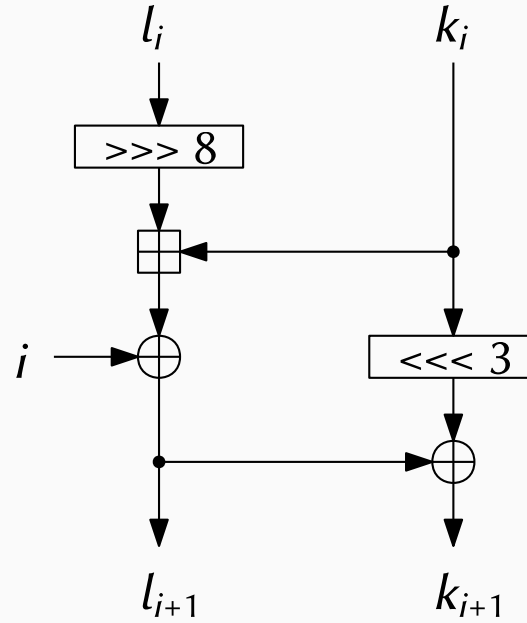
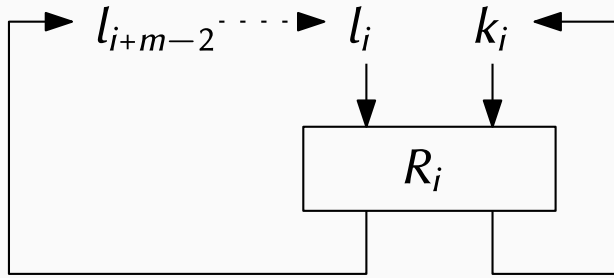
- published by NSA (2013)
 - controversy with ISO standardization, Linux kernel inclusion etc.
- family of variants with various block and key sizes
 - excellent performance in HW and SW
 - optimized for software, ARX cipher (modular addition, rotation, and XOR)
 - no realistic attacks known
- 10 variants of block/key lengths
 - the smallest: 32-bit block and 64-bit key (22 internal rounds)
 - the largest: 128-bit block with 128, 192, or 256-bit key (32, 33, 34 rounds)
- NIST selected Ascon family of algorithm as a lightweight standard (2023)
 - not a block cipher; multiple algorithms: AEAD, hash, XOF, CXOF

- round function
- input/output:
 $2n$ -bit block (two n -bit words)
- round key k_i



Speck – key expansion

- a key $K = (l_{m-2}, \dots, l_0, k_0)$ consists of m words, $m \in \{2, 3, 4\}$, $m = |K|/n$
 - for example: $m = 2$ for Speck128/128, $m = 4$ for Speck128/256
- round function is used for key expansion



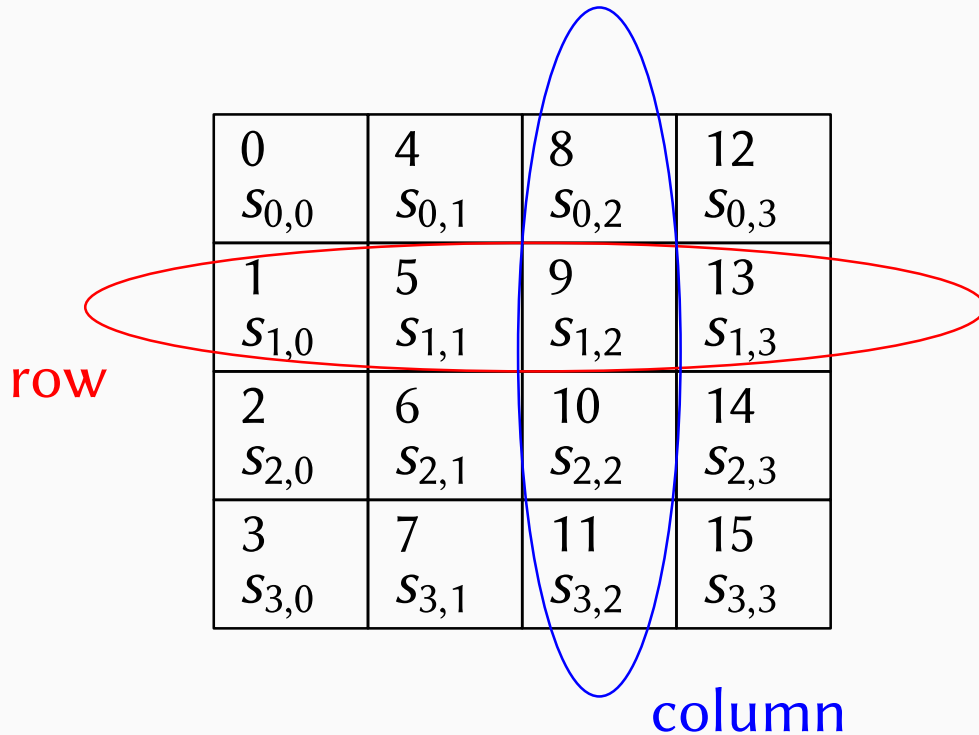
AES (Advanced Encryption Standard)}

- previous standard: DES
 - short key length (56 bits), short block length (64 bits)
- public standardization process for a new encryption standard (1997–2000)
- requirements: block cipher, block length 128 bits, key lengths 128, 192, 256 bits
- Rijndael – chosen algorithm (Vincent Rijmen, Joan Daemen)
- NIST standardized AES in 2001 (other standardizations followed)
- the most important symmetric cipher today
- used (almost) everywhere

- **not** a Feistel cipher
- different number of rounds depending on key length:
AES-128 10 rounds, AES-192 12 rounds, AES-256 14 rounds
- slight performance degradation for longer key lengths
- modern processors support AES-NI instruction set (HW accelerated AES)

AES – state and internal operations

State (plaintext, internal state, ciphertext)
4 × 4 array of bytes



The diagram shows a 4x4 state matrix. A red oval labeled 'row' highlights the second row (index 1). A blue oval labeled 'column' highlights the third column (index 2). The matrix contains indices 0-3 for rows and 4-7 for columns, and corresponding state byte labels $s_{i,j}$.

0 $s_{0,0}$	4 $s_{0,1}$	8 $s_{0,2}$	12 $s_{0,3}$
1 $s_{1,0}$	5 $s_{1,1}$	9 $s_{1,2}$	13 $s_{1,3}$
2 $s_{2,0}$	6 $s_{2,1}$	10 $s_{2,2}$	14 $s_{2,3}$
3 $s_{3,0}$	7 $s_{3,1}$	11 $s_{3,2}$	15 $s_{3,3}$

Internal operations (invertible)

- AddRoundKey – XOR the state with 128-bit round key
- SubBytes – replace each byte using a fixed permutation (S-box)
- ShiftRows – cyclically shift each row of the state
- MixColumns – multiply each column by a fixed matrix

AddRoundKey:

- fast mix of a round key into a state
- XOR, self-inverse

SubBytes:

- $s_{i,j} = S(s_{i,j})$ for all $0 \leq i, j \leq 3$
- the only nonlinear operation in AES
- carefully chosen (linear/affine ciphers are easy to break)
- invertible: inverse permutation on $\{0, 1\}^8$

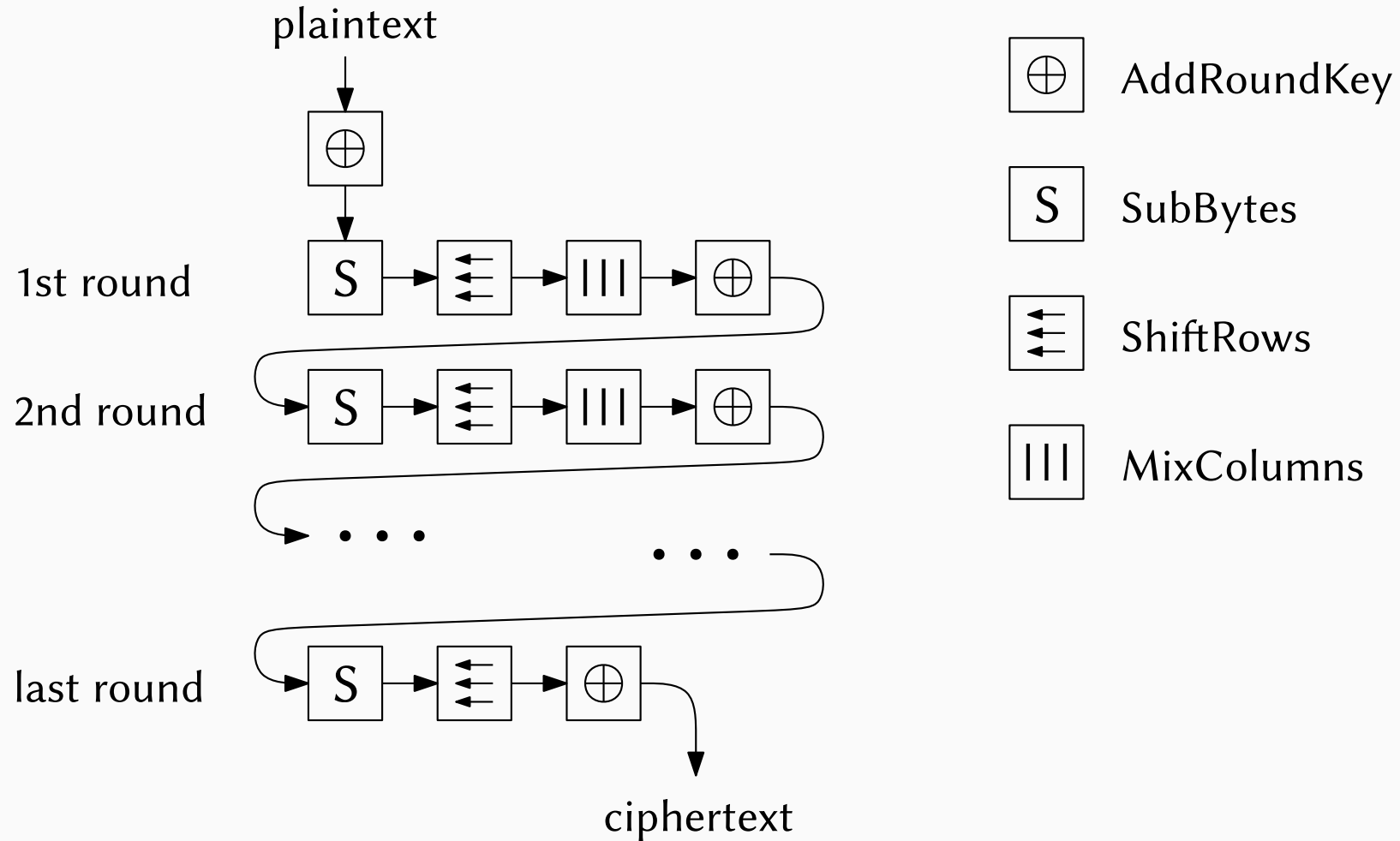
ShiftRows

- 1st row is not shifted
- 2nd/3rd/4th row: bytes are cyclically shifted to the left by 1/2/3 bytes
- example: $(s_{1,0}, s_{1,1}, s_{1,2}, s_{1,3}) \mapsto (s_{1,1}, s_{1,2}, s_{1,3}, s_{1,0})$
- invertible: shift to the right

MixColumns

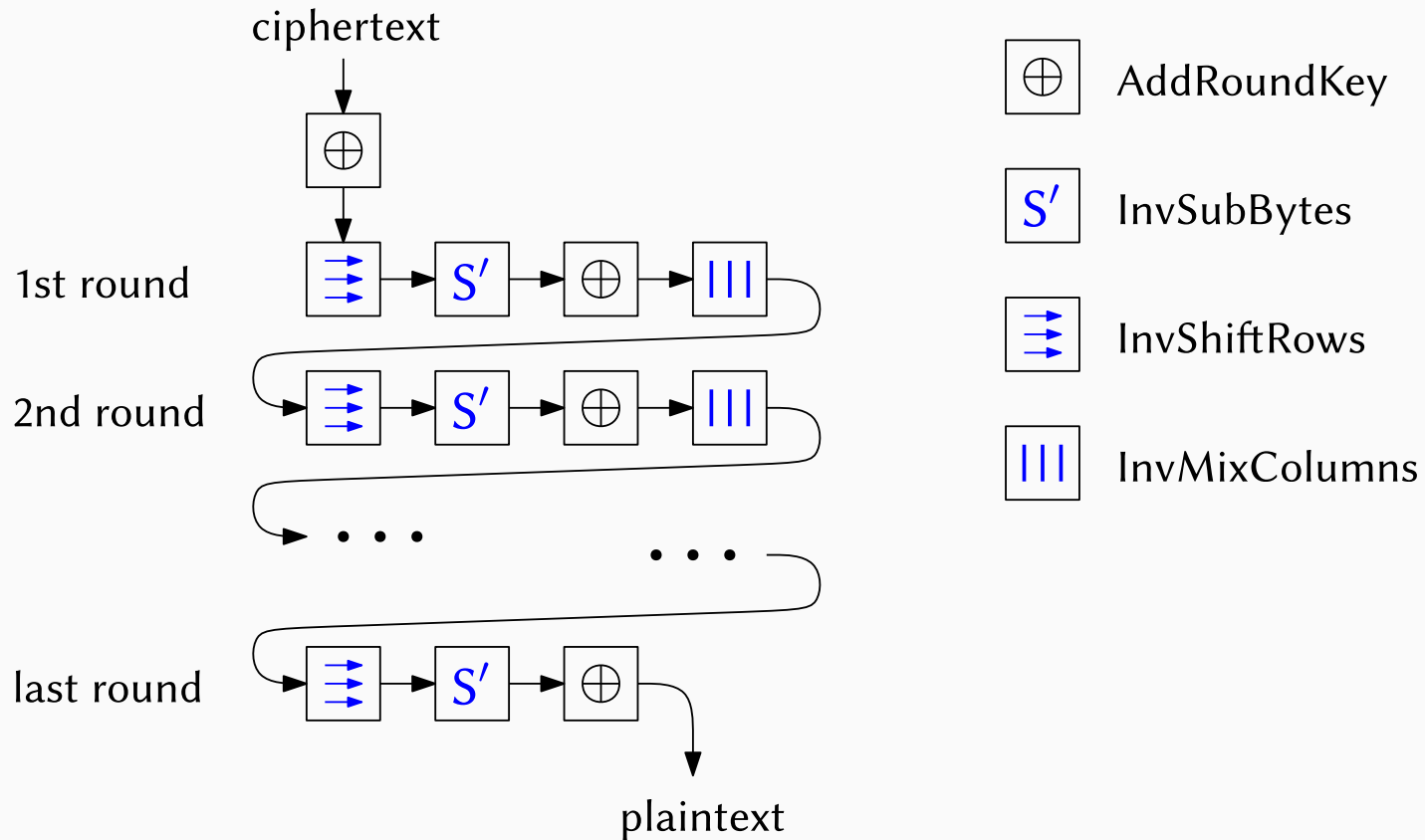
- fixed (invertible!) matrix M
- good diffusion properties (small difference on input is “amplified”)

AES – encryption



AES – decryption

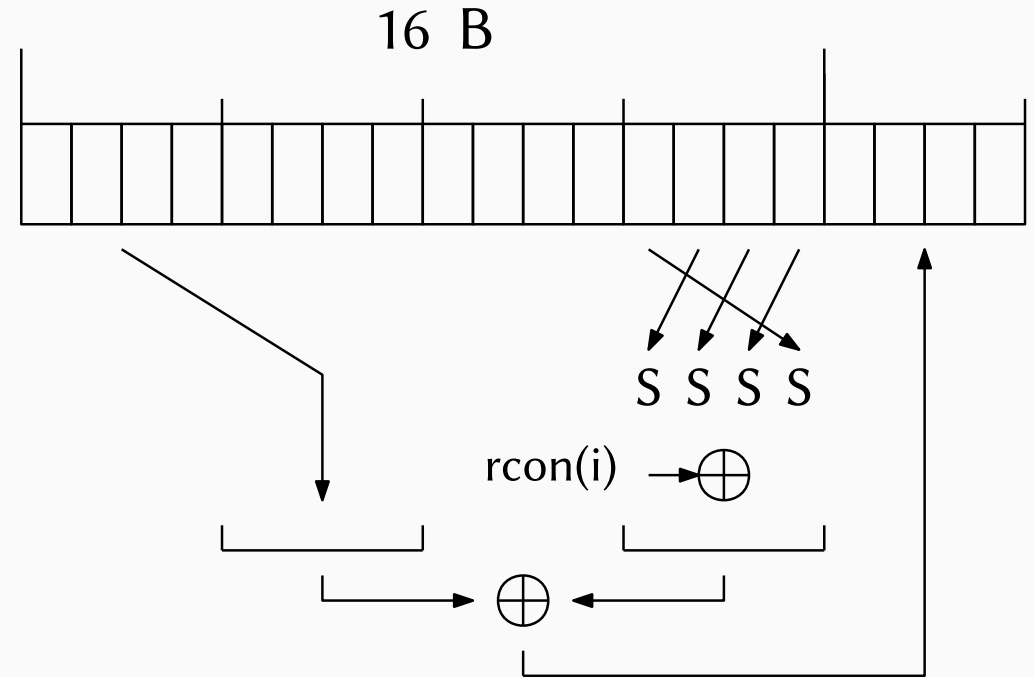
inverse operations: InvShiftRows, InvMixColumns, InvSubBytes



AES – key expansion for 128 bit key

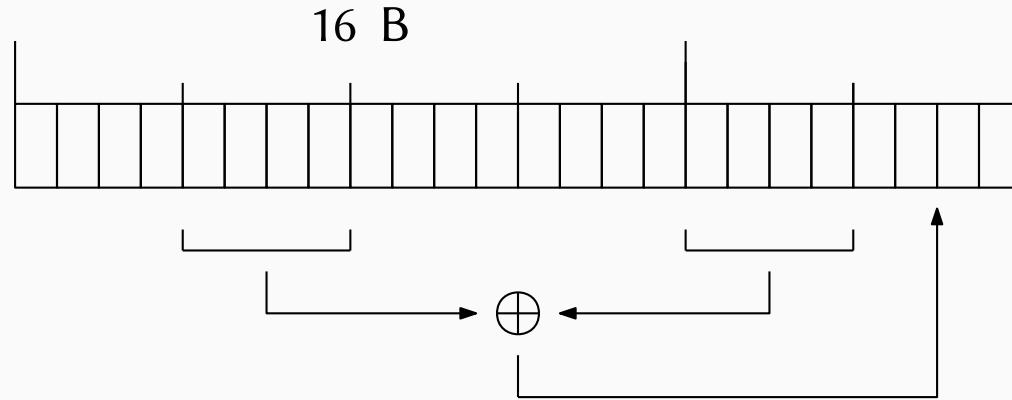
- AES-128 \Rightarrow 10 rounds \Rightarrow 11 round keys
($11 \cdot 16 = 176$ bytes)
- first 16 bytes (first round key) is the encryption key
- $\text{rcon}(i)$ – round constant

1st 4-byte word in each new round key:



AES – key expansion for 128 bit key (cont.)

- for the 2nd, 3rd, and 4th 4-byte word in each round key:



- round keys are formed from consecutive bytes of the expanded key
- slightly different key expansion for key length 256

- brute force complexity:
 2^{128} or 2^{192} or 2^{256}
- best key recovery attacks
 - Tao and Wu (2015),
KPA:

	time	data
AES-128	$2^{126.1}$	2^{56}
AES-192	$2^{189.9}$	2^{48}
AES-256	$2^{254.3}$	2^{40}

- security of reduced AES
- key recovery attacks of 7-round AES-128

Attack	Rounds	Data	Time	Memory	Key schedule
Impossible Differential	7	$2^{112.2}$	$2^{117.2}$	$2^{112.2}$	yes
Meet-in-the-Middle	7	2^{116}	2^{116}	2^{116}	yes
Impossible Differential	7	$2^{105.1}$	2^{113}	$2^{74.1}$	yes
Impossible Differential	7	$2^{104.9}$	$2^{110.9}$	$2^{71.9}$	yes
Zero-Difference	7	$2^{110.2}$	$2^{110.2}$	$2^{110.2}$	no
Meet-in-the-Middle	7	2^{97}	2^{99}	2^{98}	yes

source: <https://eprint.iacr.org/2022/487.pdf>

Multiple encryption

- multiple encryption (cascade encryption)
 - using the same or different ciphers, usually with independent keys
 - two ciphers cascade: $E_{k_1, k_2}(p) = E'_{k_2}(E^*_{k_1}(p))$
- possible goals:
 - increasing the key space
 - in case one cipher is broken ... use two or three distinct
- some ciphers cannot be strengthened regardless of cascade length
 - the key space does not increase
 - examples: simple substitution, Vigenere, permutation, Vernam, etc.
 $\forall k_1, k_2 \exists k \forall p : E_{k_2}(E_{k_1}(p)) = E_k(p)$
- independence of keys can be crucial
 - example: using the same key in double Vernam cipher \Rightarrow no encryption

TDEA (3DES)

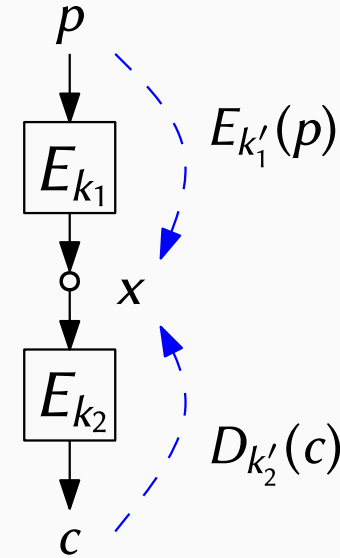
- 3DES is defined as a cascade of length 3:
 - encryption: $E_{k_3}(D_{k_2}(E_{k_1}(p)))$
 - decryption: $D_{k_1}(E_{k_2}(D_{k_3}(c)))$
- keying options and the corresponding key length:
 - option 1: independent keys (168 bits)
 - option 2: $k_1 = k_3$ (112 bits)
 - option 3: $k_1 = k_2 = k_3$ (56 bits)
- EDE mode (instead of EEE mode) and keying option 3 ensures backward compatibility with DES
- real strength (bit security) of 3DES:
 - option 1: 112 bits (meet in the middle attack)
 - option 2: 80 bits (assuming 2^{40} known plaintext/ciphertext pairs)

Meet in the middle attack (MITM)

- disadvantage of multiple encryption – slower than single encryption
- Why not “double encryption”? → MITM attack!
 - MITM is generally applicable to multiple encryption schemes
 - MITM is known plaintext attack (several pairs (p_i, c_i) are known)

$$c = E_{k_2}(E_{k_1}(p))$$

1. $\forall k'_2$: compute $x = D_{k'_2}(c)$ and store (x, k'_2) in a hash table indexed by x
2. $\forall k'_1$: compute $x = E_{k'_1}(p)$
 - find entry(ies) (x, k'_2) in the table
 - verify a candidate key(s) (k'_1, k'_2) using other plaintext/ciphertext pairs



MITM – complexity

- assume key length k and block length n
- expected number of required plaintext/ciphertext pairs is $\lceil 2k/n \rceil$
 - $\approx 2^{2k}/2^n$ “valid” key pairs for a single (p, c) pair
 - $\approx 2^{2k}/2^{tn}$ for t plaintext/ciphertext pairs
 - from $1 \approx 2^{2k}/2^{tn}$ we get $t \approx 2k/n$
- time complexity $O(2^k)$
 - first cycle 2^k iterations; second cycle 2^k iterations
 - single hash table operation $O(1)$
- memory complexity $O(2^k)$
 - each key k'_2 produces one fixed-length entry in the hash table
 - second cycle in constant memory
- easily generalized for longer cascades
 - example: MITM on 3DES with 3 keys – time 2^{112} and memory 2^{56}

A KPA on two-key triple encryption

- example cipher: 3DES with keying option 2, $c = E_{k_1}(D_{k_2}(E_{k_1}(p)))$
- slightly more involved than MITM attack on double-encryption
- assume t known plaintext/ciphertext pairs
- time complexity: $O(t + 2^{k+n-\lg t})$, memory complexity: $O(t + 2^{k-n} \cdot t)$
- 3DES with two key option:
 - parameters: $k = 56, n = 64, t = 2^{40}$
 - time complexity: $O(t + 2^{k+n-\lg t}) \approx 2^{120-40} = 2^{80}$
 - memory complexity: $O(t + 2^{k-n} \cdot t) \approx 2^{40}$
- Triple AES-128 (not used in practice) with two-key option:
 - parameters: $k = 128, n = 128, t = 2^{60}$
 - time / memory complexity: $\approx 2^{196} / \approx 2^{60}$
- different trade-offs for different t values

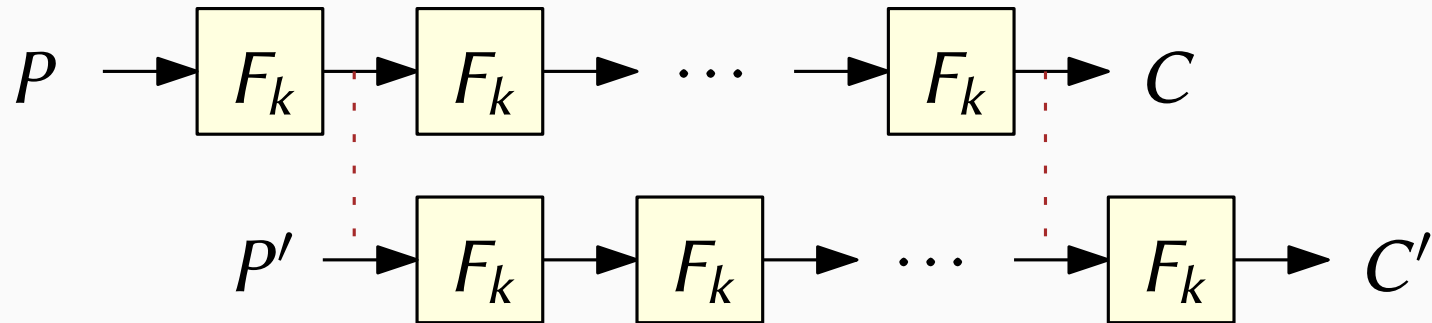
Data requirements of KPA/CPA

- assumption: block length $n = 128$
- only the ciphertext is considered for size computation, and for calculation of transmission time

data	size [TB]	time for 1Gb/s
2^{40}	17.6	39 hours
2^{60}	$1.8 \cdot 10^8$	4676 years
2^{80}	$1.9 \cdot 10^{13}$	$4.9 \cdot 10^9$ years
2^{100}	$2.0 \cdot 10^{19}$	$5.1 \cdot 10^{15}$ years

- iterated ciphers
 - easy to change the number of rounds
 - usually more rounds \approx increased security
 - key scheduling is important
- Biryukov, Wagner (1999)
 - general attack on iterated cipher with identical round transform
 - arbitrary number of rounds
 - other variants exist
- cipher: $C = F_k \circ F_k \circ \dots \circ F_k(P)$

- *slid pair* is a known pair (P, C) and (P', C') such that $P' = F_k(P)$ and $C' = F_k(C)$



Slide attack – how

- we assume that F_k is “weak”:
 - easy to compute k from equations $y_0 = F_k(x_0), y_1 = F_k(x_1)$
 - usually very easy; for example, try this for Speck2n or AES
- KPA attack
 - approx. $2^{n/2}$ of known plaintext-ciphertext pairs
⇒ expecting ≈ 1 slid pair (birthday paradox)
 - testing all combinations if there is a slid pair $(P, C), (P', C')$
Is there k such that $P' = F_k(P) \wedge C' = F_k(C)$? ... ($\approx 2^n$)
 - one slid pair recovers approx. n bits of the key
- Why bother when time complexity is $O(2^n)$?
 - single round (slide attack) vs. full cipher (brute-force)
 - other improvements depending on F

- CPA slide attacks much better with Feistel ciphers
 - single round ... half of the block does not change
 - $\approx 2^{n/4}$ plaintext-ciphertext pairs for finding a slid pair
 - complexity is $O(2^{n/2})$
- advanced variants of slide attack exist
- pay attention to key scheduling

1. Decrypt an ASCII plaintext block encrypted using AES-128 and a key in the form $b'0000000000?????'$, where ? are some digits. The ciphertext (two representations):
 - (hex) 090db742e1ff338013701602ea2ea422
 - (bytes) $b' \backslash t \backslash r \backslash x b 7 B \backslash x e 1 \backslash x f f 3 \backslash x 8 0 \backslash x 1 3 p \backslash x 1 6 \backslash x 0 2 \backslash x e a . \backslash x a 4 " '$
2. Assess the security of AES-128, where we omit all operations
 - a) AddRoundKey
 - b) ShiftRows
 - c) MixColumns
3. Assume Speck128/128 ($n = 64$) with equal round keys. Show how to find a slid pair for this cipher efficiently in CPA scenario. Estimate the complexity of the slide attack.