Stream and Block Ciphers, DES, AES

27 March 2025
Lecture 2

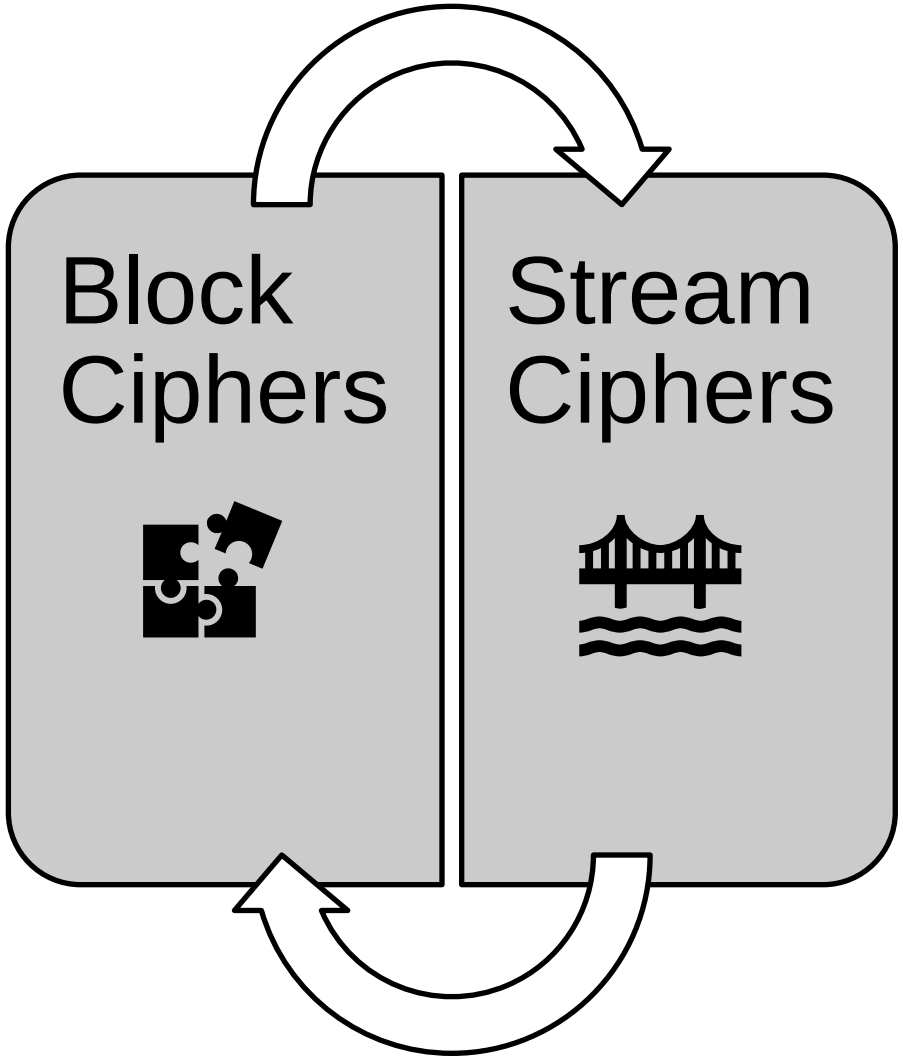
Topics for Today

- Modern Cryptographic Tools
  - Stream Ciphers
  - Block Ciphers
- Shared Key Encryption
  - DES
  - AES

Types of Ciphers

Both:

Combine input plaintext and key to produce cipher text



Differ:

How the plaintext and key are combined

Stream Ciphers

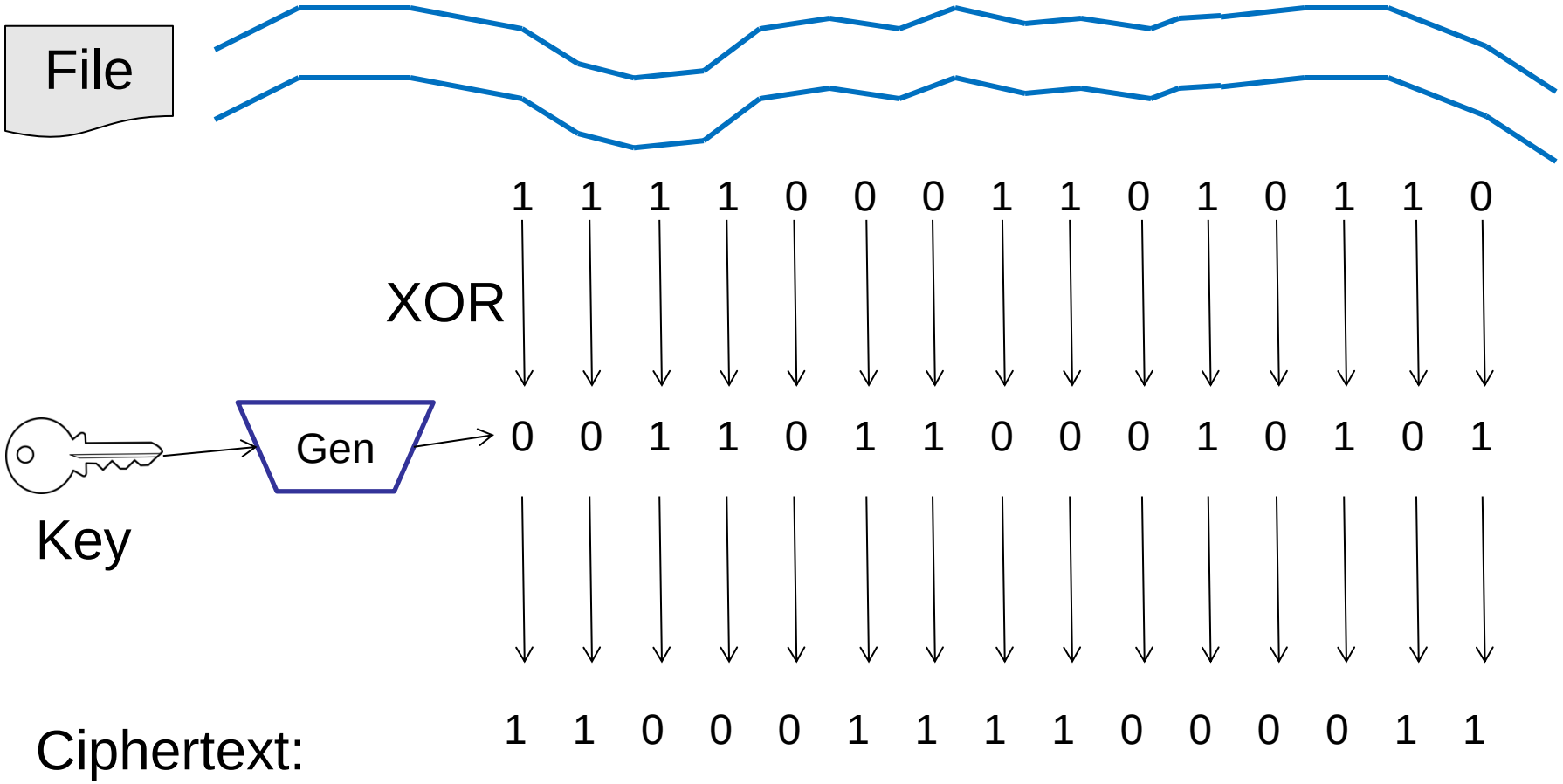
- Characterized by operating on **one symbol at a time**
  - The alphabetic substitutions we have seen so far have been stream ciphers

- If the algorithm is XOR, this is a stream cipher:

$p:$	p_1	p_2	p_3	p_4	p_5
	$\oplus$	$\oplus$	$\oplus$	$\oplus$	$\oplus$
$k:$	k_1	k_2	k_3	k_4	k_5
$c:$	c_1	c_2	c_3	c_4	c_5

- Some kinds of errors will affect subsequent encryptions and decryptions
  - The message will decrypt to a point and then fail
  - The recipient can recover by duplicating the error

Stream Imagined

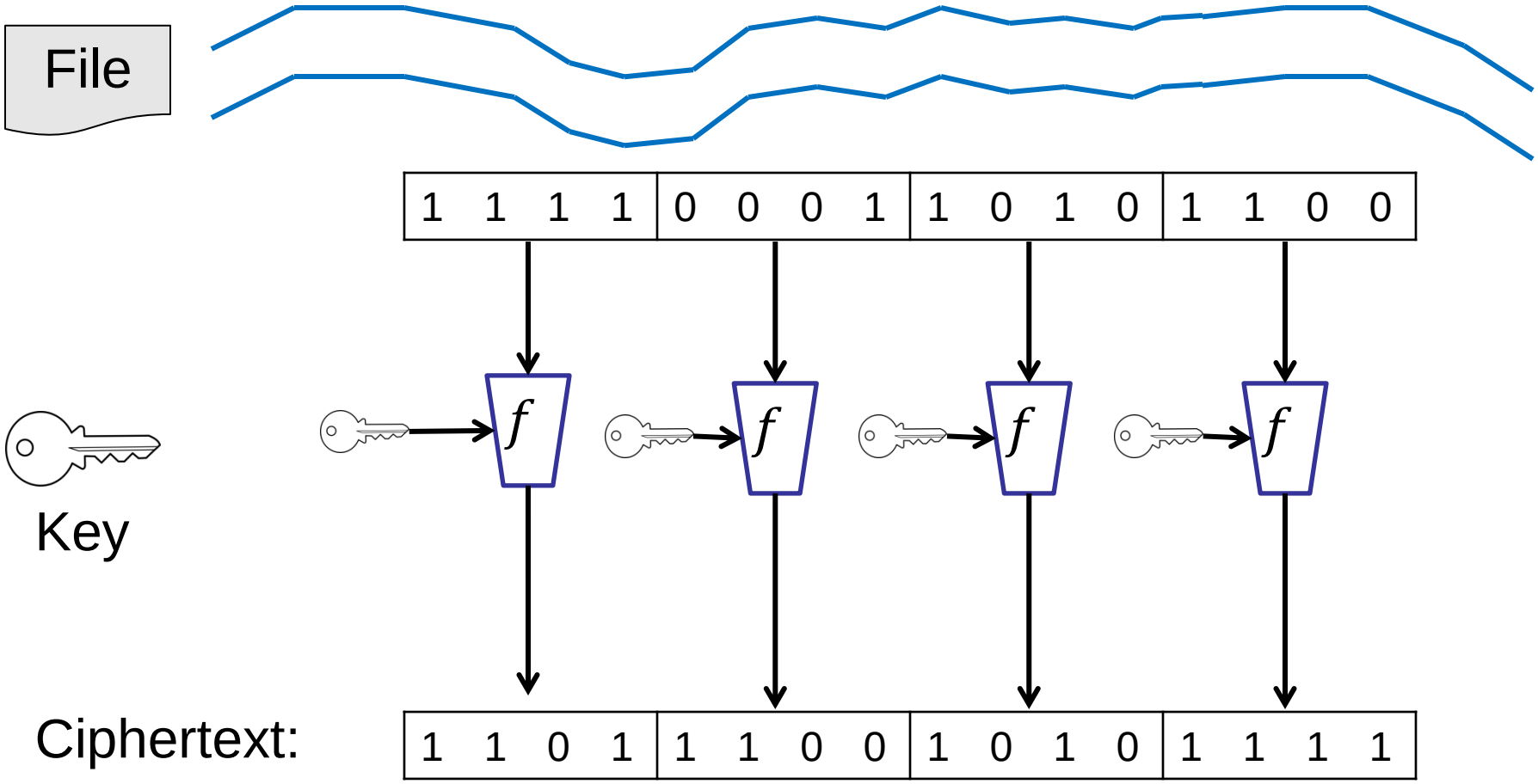


Block Ciphers

- Characterized by operating on **more than one symbol at a time**
- Takes a group of symbols as input, combines them with a secret, and outputs a block of ciphertext
  - **In a good block cipher, all of the input characters influence all of the output**
  - **Can introduce more randomness**
- If the algorithm is f, this is a block cipher:

p:	p_1p_2	p_3p_4	p_5p_6	p_7p_8	p_9p_{10}
	f	f	f	f	f
k:	k_1	k_2	k_3	k_4	k_5
c:	c_1c_2	c_3c_4	c_5c_6	c_7c_8	c_9c_{10}

Block Imagined



Advantages and Disadvantages

Stream ciphers:

- Operate relatively fast since they work on only one character at a time
- Lower error propagation since each symbol is affected only by itself

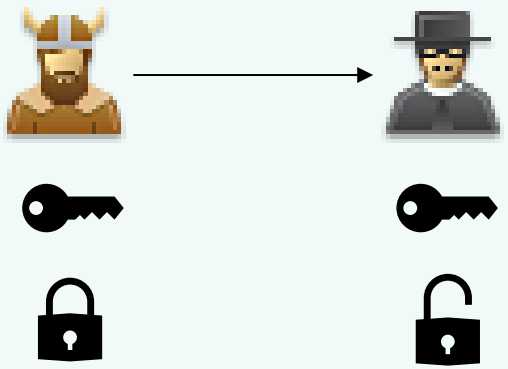
Block Ciphers:

- Higher diffusion since the material for each block affects the entire block
- Single characters can not be swapped in and out by an attacker

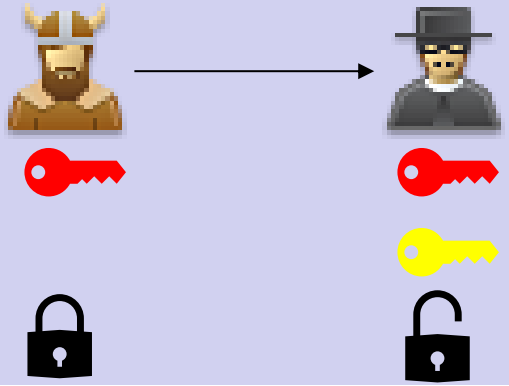
What about mapping one symbol to multiple symbols?

Kinds of Industrial Strength Crypto

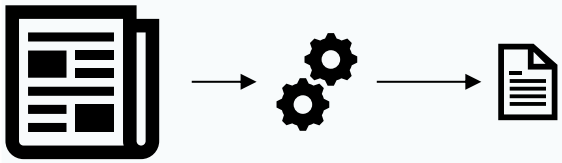
Shared Key Cryptography



Public Key Cryptography



Cryptographic Hashes



- All aim for computational security
- Not all methods have been proved to be intractable to crack.

Shared Key/Symmetric Cryptography

- Sender & receiver use same (secret) key
  - *Block* or *stream ciphers*
  - Ex: (Block): DES, Triple-DES, Blowfish, AES

Encryption E : $key\ x\ plain \rightarrow cipher$

Decryption D : $key\ x\ cipher \rightarrow plain$

D inverts E

$D(K, E(K, msg)) = msg$

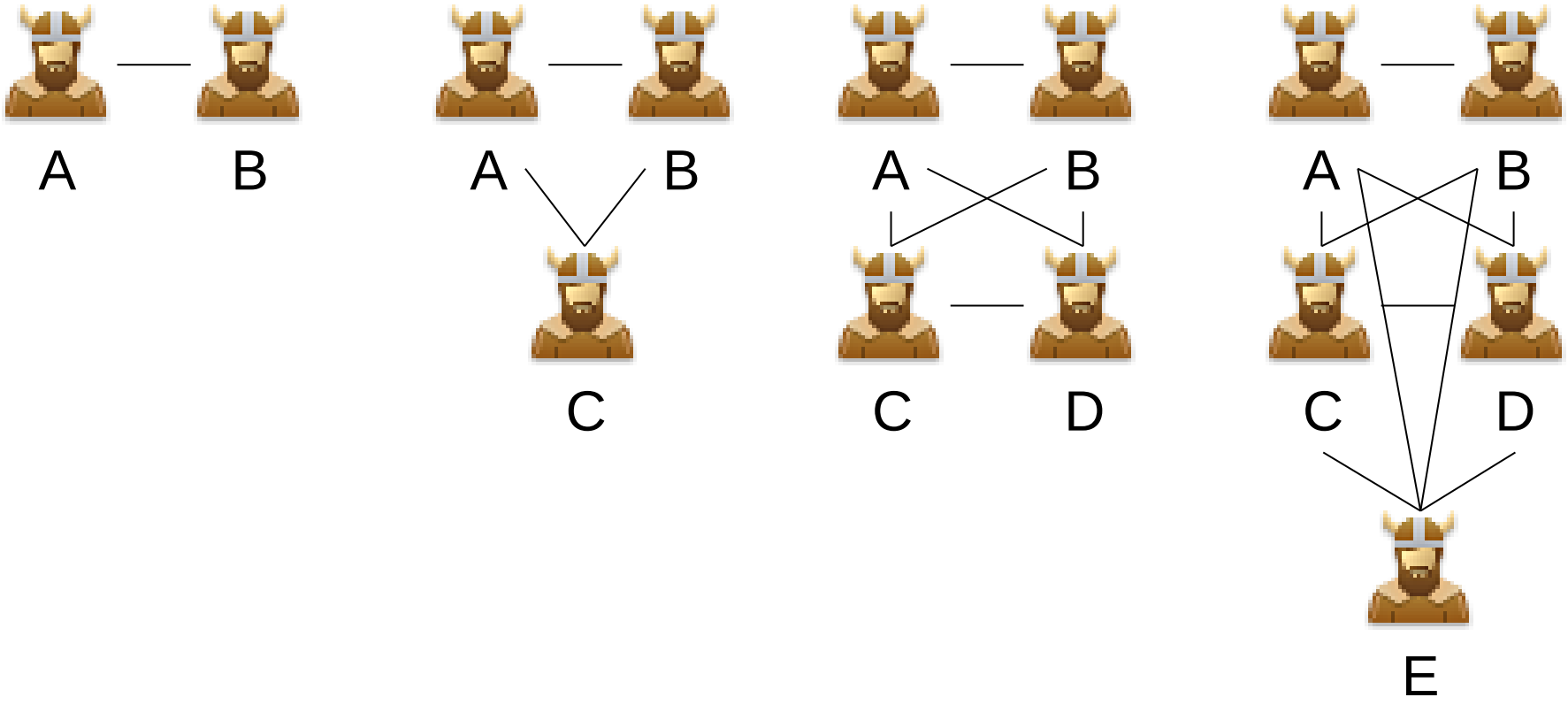
Sometimes E is same algorithm as D

Challenges to Shared Key Crypto

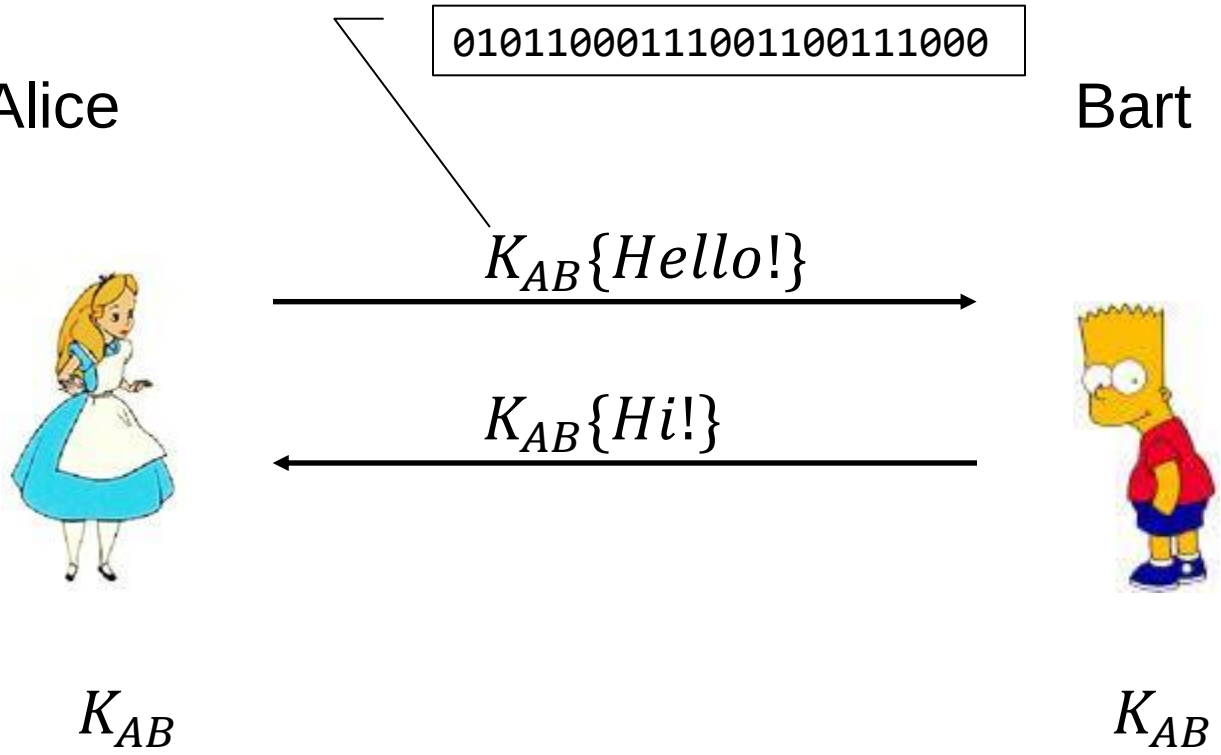
- Stolen key means attacker can decrypt old ciphertext
  - → Change keys frequently to limit damage

- Distribution of keys is problematic
  - Keys must be transmitted securely
  - Use couriers?
  - Distribute in pieces over separate channels?
- Number of keys is $O(n^2)$ where n is # of participants

Pairs of Shared Keys



Secure Channel: Shared Keys



An outsider shouldn't be able to tell:

1. The message contents
2. Which key the message was encrypted under

Data Encryption Standard (DES)

Security analyzed by National Security Agency (NSA)

- Standardized 1976
- http://csrc.nist.gov/publications/fips/fips46-3/fips46-3.pdf

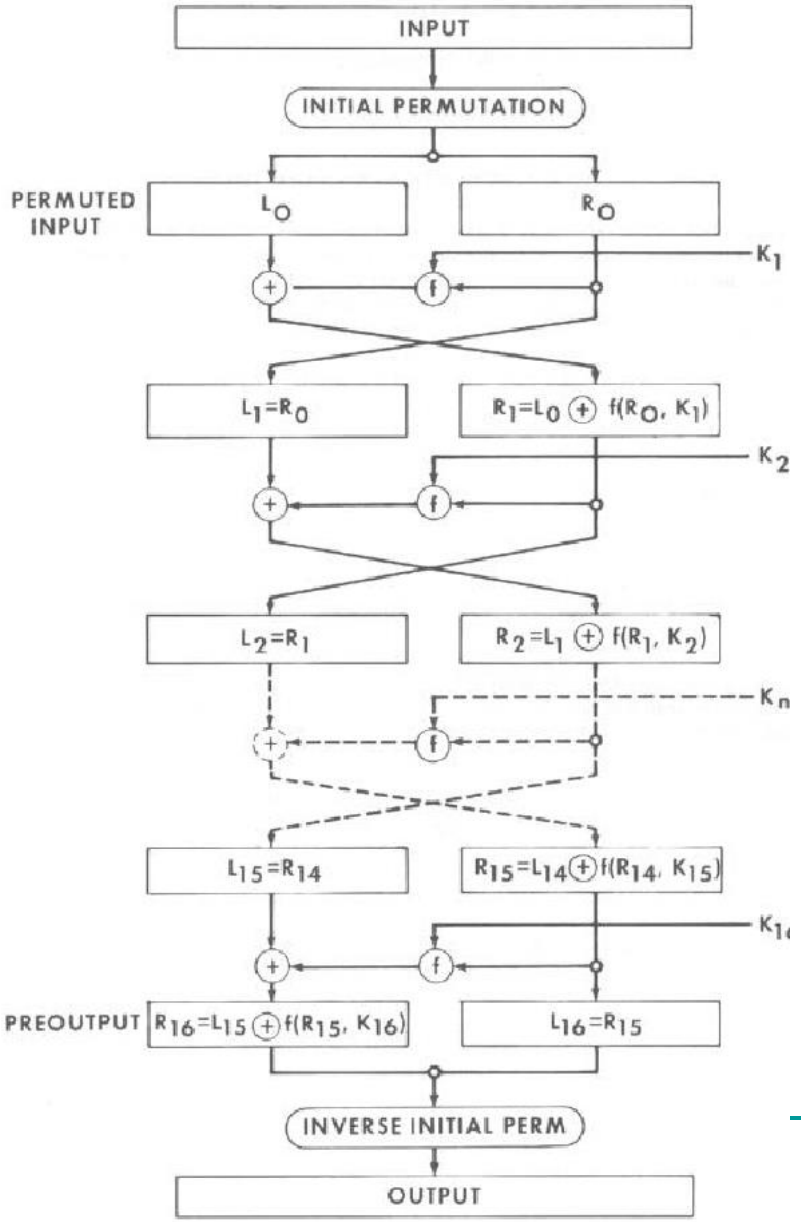
Key is 56b

- Padded to 64b using 8 parity bits
- Input processed in 64b blocks

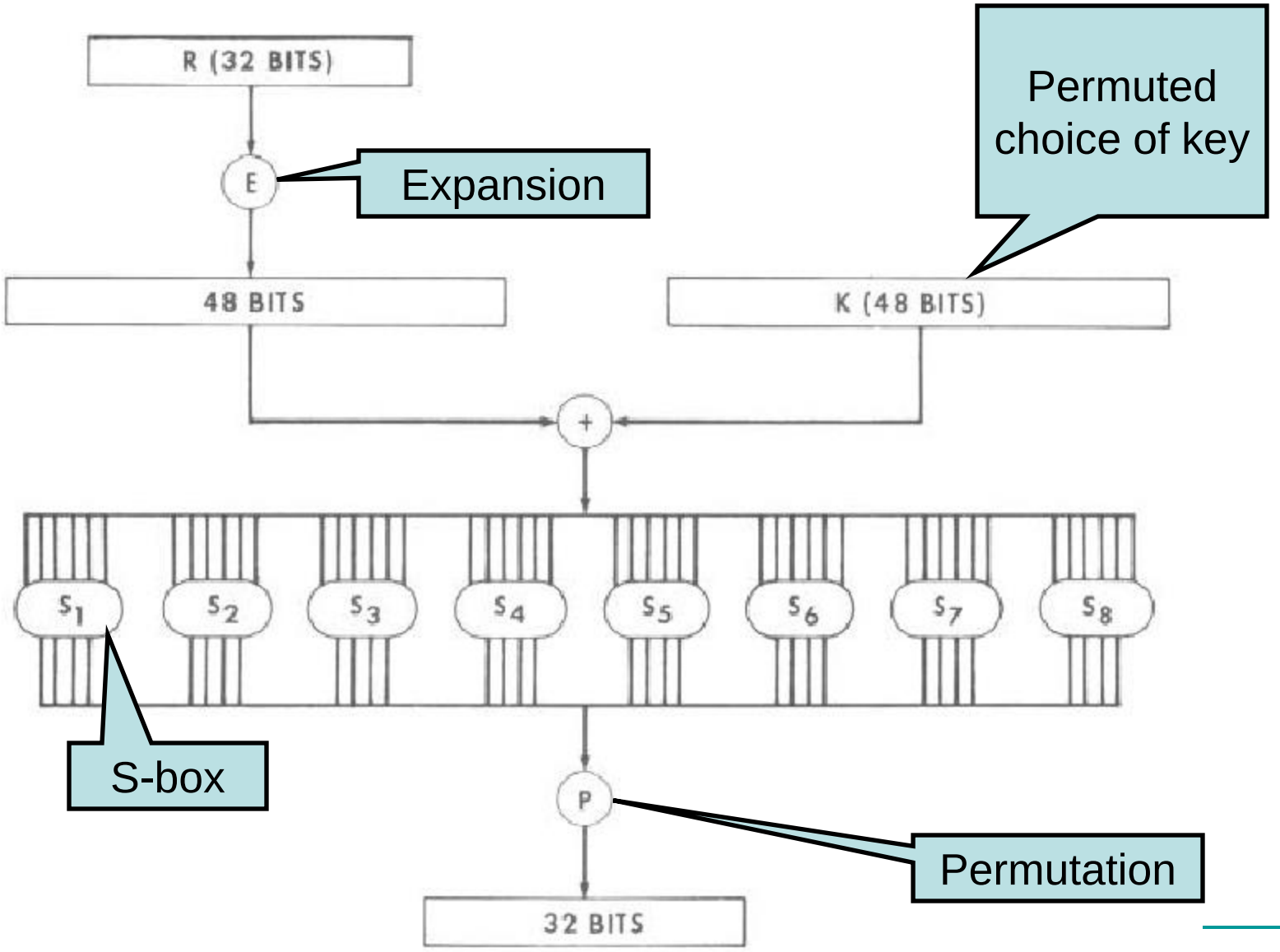
Uses simple operators on (up to) 64b values

- Simple to implement in software or hardware
- Based on a series of 16 *rounds*
- Each cycle uses permutation & substitution to combine plaintext with the key

DES Encryption



One Round of DES (f of previous slide)



DES S-Boxes

- Substitution table
- 6 bits of input replaced by 4 bits of output
- Which substitution is applied depends on the input bits

- Implemented as a lookup table
  - 8 S-Boxes
  - Each S-Box has a table of 64 entries
  - Each entry specifies a 4-bit output

שורה	מס' עמודה															
	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	S_1															
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	3	14	2	13	1	10	6	12	11	9	5	3	8
2	4	1	14	8	13	6	2	11	15	12	9	7	13	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
	S_2															
0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
2	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
3	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
	S_3															
0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
2	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
3	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
	S_4															
0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
2	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
	S_5															
0	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
1	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
2	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
3	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
	S_6															
0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
2	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
3	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
	S_7															
0	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
1	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
2	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
3	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
	S_8															
0	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
2	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
3	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

DES Decryption

- Use the same algorithm as encryption, but use $k_{16} \ldots k_1$ instead of $k_1 \ \ldots \ k_{16}$
- Proof that this works:
  - To obtain round j from $j-1$:

    (1) $L_j = R_{j-1}$

    (2) $R_j = L_{j-1} \oplus f(R_{j-1}, k_j)$
  - Rewrite in terms of round $j-1$:

    1) $R_{j-1} = L_j$

    2) $L_{j-1} \oplus f(R_{j-1}, k_j) = R_j$

    3) $L_{j-1} \oplus f(R_{j-1}, k_j) \oplus f(R_{j-1}, k_j) = R_j \oplus f(R_{j-1}, k_j)$

    4) $L_{j-1} = R_j \oplus f(R_{j-1}, k_j)$

    5) $L_{j-1} = R_j \oplus f(L_j, k_j)$

Problems with DES

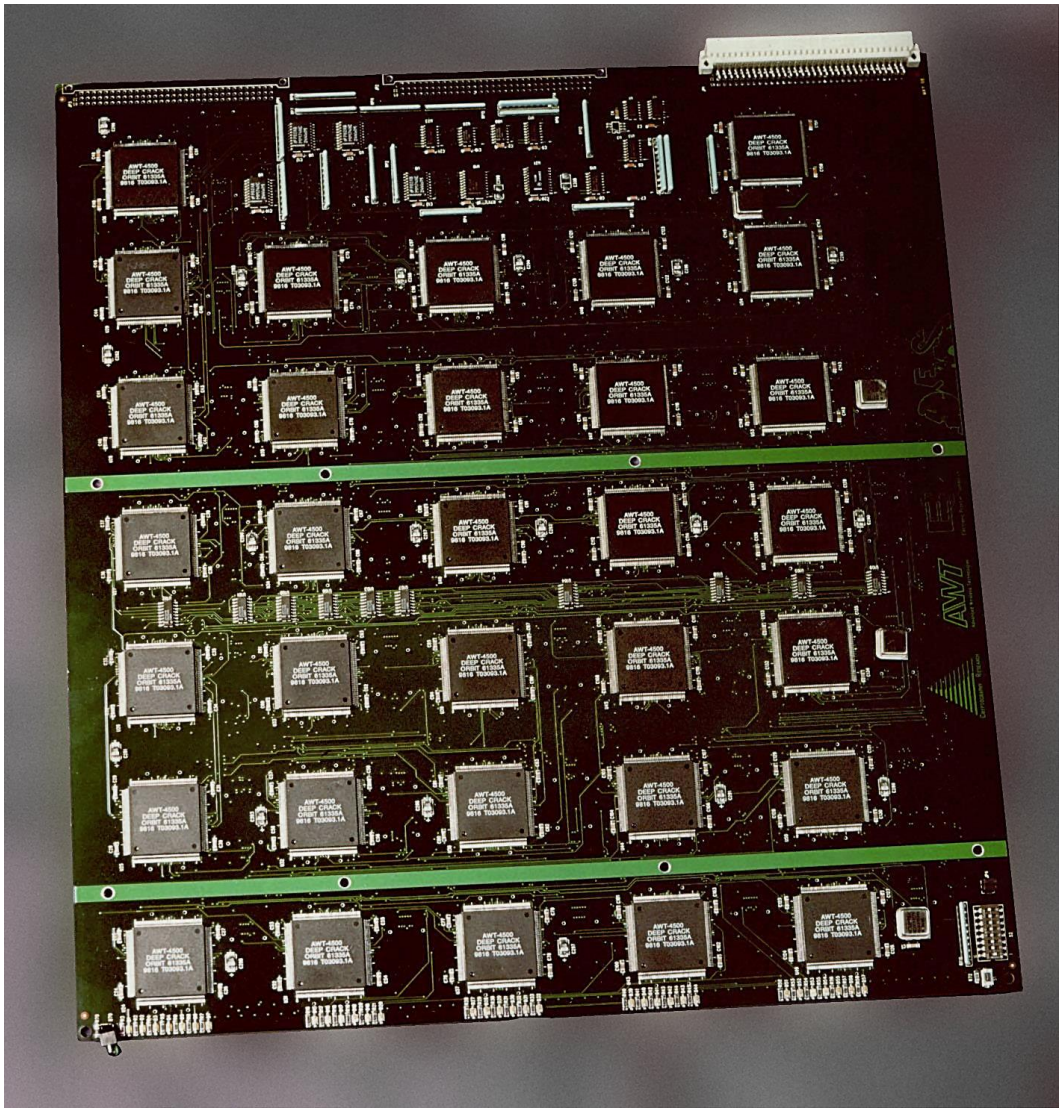
- Key length too short: 56 bits
  - www.distributed.net broke a DES challenge in 1999 in under 24 hours (parallel attack)
- Other problems
  - Bit-wise complementation of key and message produces bit-wise complemented ciphertext
  - Not all keys are good (Weak Keys)
    - `0x0101010101010101, 0xFEFEFEFEFEFEFEFE, 0xE0E0E0E0F1F1F1F1, 0x1F1F1F1F0E0E0E0E`
  - Semi-weak keys (they are swappable)

First key in the pair	Second key in the pair
01FE 01FE 01FE 01FE	FE01 FE01 FE01 FE01
1FE0 1FE0 0EF1 0EF1	E01F E01F F10E F10E
01E0 01E0 01F1 01F1	E001 E001 F101 F101
1FFE 1FFE 0EFE 0EFE	FE1F FE1F FE0E FE0E
011F 011F 010E 010E	1F01 1F01 0E01 0E01
E0FE E0FE F1FE F1FE	FEE0 FEF0 FEF1 FEF1

Be careful if you choose keys randomly!

https://publicdomainvectors.org/en/free-clipart/Weak-man-drinking/65657.html

DES Cracker



By The original uploader was Matt Crypto at English Wikipedia. Later versions were uploaded by Ed g2s at en.wikipedia. - http://w2.eff.org/Privacy/Crypto/Crypto_misc/DESCracker/ (Via en:)., CC BY 3.0 us, https://commons.wikimedia.org/w/index.php?curid=2437815

Attacks on DES

Differential cryptanalysis (1990)

Carefully choose pairs of plaintext that differ in particular known ways (e.g. they are complements)

- But particular choice of S boxes is secure against this (!)
- Shamir and Biham found an attack with 2^{47} chosen plaintexts

Linear cryptanalysis (1994)

Convert the cipher to a series of linear equations and analyze using chosen plaintexts

- Matsui found an attack with 2^{43} chosen plaintexts (surprised NSA!)

Triple-DES

- DES was attackable – but still only by brute force.
- How to improve? Increase the key length? 😟
- Many hardware implementations of DES existed and it had been basically proven secure

- Why not encrypt it twice?

- Some options:
- $E(k_1, E(k_1, plaintext))$
  - Doesn't increase the key space at all! Why?

- $E(k_2, E(k_1, plaintext))$
  - Interestingly only doubles the key space (same as 2^{57}, not what we want)

Triple-DES: Solution

$$E(k_1, D(k_2, E(k_1, plaintext)))$$

- Encrypt with k_1, decrypt with k_2, encrypt with k_1
- Increases the key space to 2^{112}, at the price of three times the operations (Triple-DES)

•Another option: $E(k_1, D(k_2, E(k_3, plaintext)))$ to give 168 bits, but be careful

- Some key interaction problems exist

So Far

- Modern Cryptographic Tools
  - Stream Ciphers
  - Block Ciphers
- Shared Key Encryption
  - DES
  - AES

Advanced Encryption Standard (AES)

- National Institute of Standards & Technology NIST
  - Computer Security Research Center (CSRC)
  - http://csrc.nist.gov/

- January 1997: NIST announces new algorithm will be chosen publicly
- September 1997: NIST releases call:
  - Must support 128, 192, 256 bits for key
  - Must support 128 bit block size
- 15 submissions over 9 months
- 5 finalists:
  - Rijndael, Serpent, Twofish, RC6, MARS

- Three AES conferences 1998-2000

- April 2000: Winner: Rijndael

Image credits: http://blog.code42.com/wp-content/uploads/2015/05/AES256-Shield3.jpg

About AES (Rijndael)

- http://www.esat.kuleuven.ac.be/~rijmen/rijndael/

Invented by Belgium researchers Dr. Joan Daemen & Dr. Vincent Rijmen

- Based on Square algorithm

Adopted May 26, 2002

Key length: 128, 192, or 256 bits

Block size: 128 (192, or 256) bits

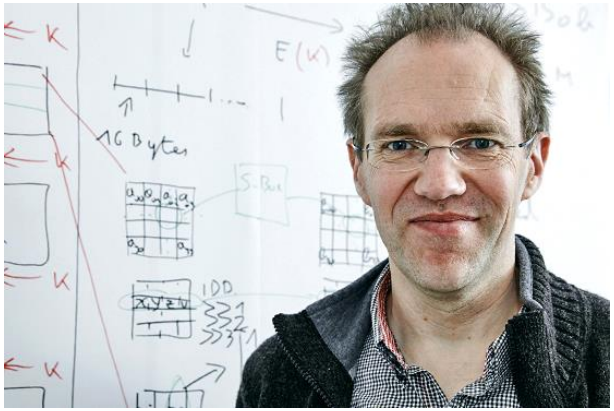


Image credits: https://www.epo.org/learning-events/european-inventor/finalists/2016/daemen/DaemenGallery1.jpg, https://www.esat.kuleuven.be/cosic/wp-content/uploads/2016/12/DSC01837.jpg

Advanced Encryption Standard (AES)

Differs from DES in that

- Variable number of rounds
- Variable key size (128, 192, 256)
- Not Feistel - works on the whole message at once
- Includes columnar transposition in addition to permutations

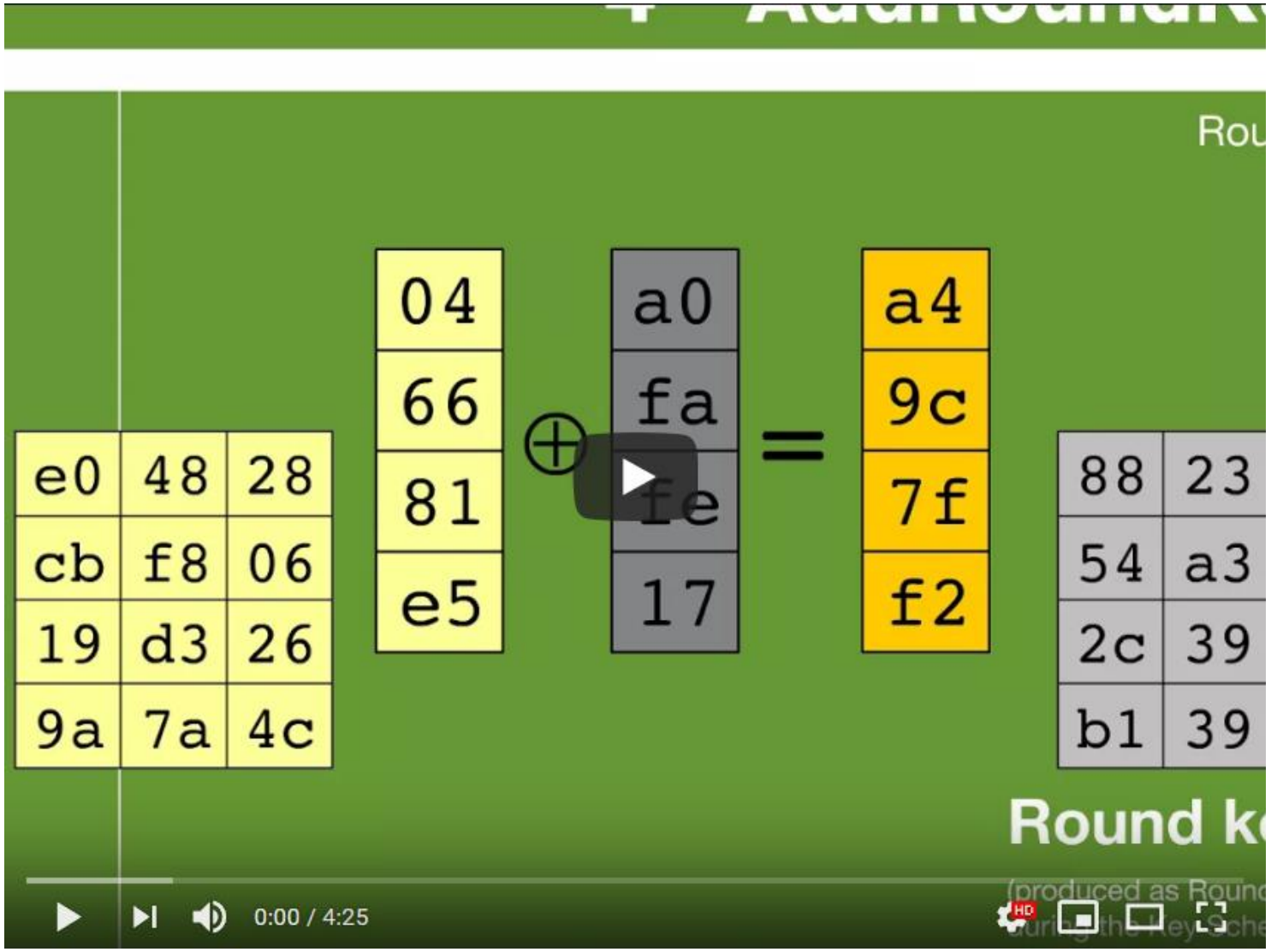
AES operations

- Substitution (SBoxes)
- Row Shifts
- Column Combinations

Has held up to public scrutiny for now

- Read more on it and how it works on Wikipedia

AES Steps

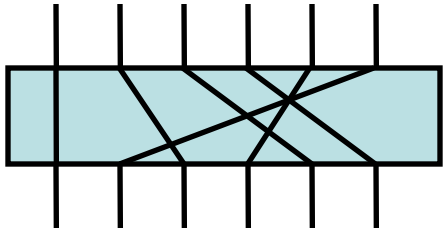

Rou
e0 48 28
cb f8 06
19 d3 26
9a 7a 4c
04
66
81
e5
a0
fa
fe
17
a4
9c
7f
f2
88 23
54 a3
2c 39
b1 39
Round k
(produced as Round
0:00 / 4:25

Conclusion

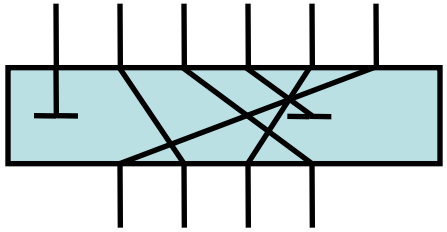
- Modern Cryptographic Tools
  - Stream Ciphers
  - Block Ciphers
- Shared Key Encryption
  - DES
  - AES

Types of Permutations in DES

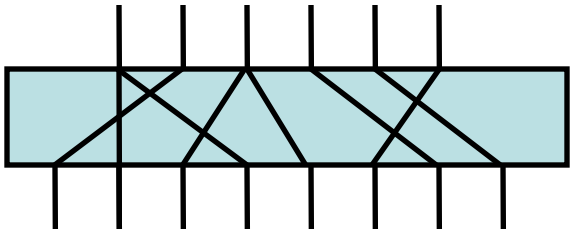
Permutation



Permuted Choice



Expansion Permutation



DES S-Box Example

S_5		Middle 4 bits of input														
		0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
Outer bits	**00**	0010	1100	0100	0001	0111	1010	1011	0110	1000	0101	0011	1111	1101	0000	1110	1001
	01	1110	1011	0010	1100	0100	0111	1101	0001	0101	0000	1111	1010	0011	1001	1000	0110
	10	0100	0010	0001	1011	1010	1101	0111	1000	1111	1001	1100	0101	0110	0011	0000	1110
	11	1011	1000	1100	0111	0001	1110	0010	1101	0110	1111	0000	1001	1010	0100	0101	0011