EE488
Introduction to Cryptography Engineering

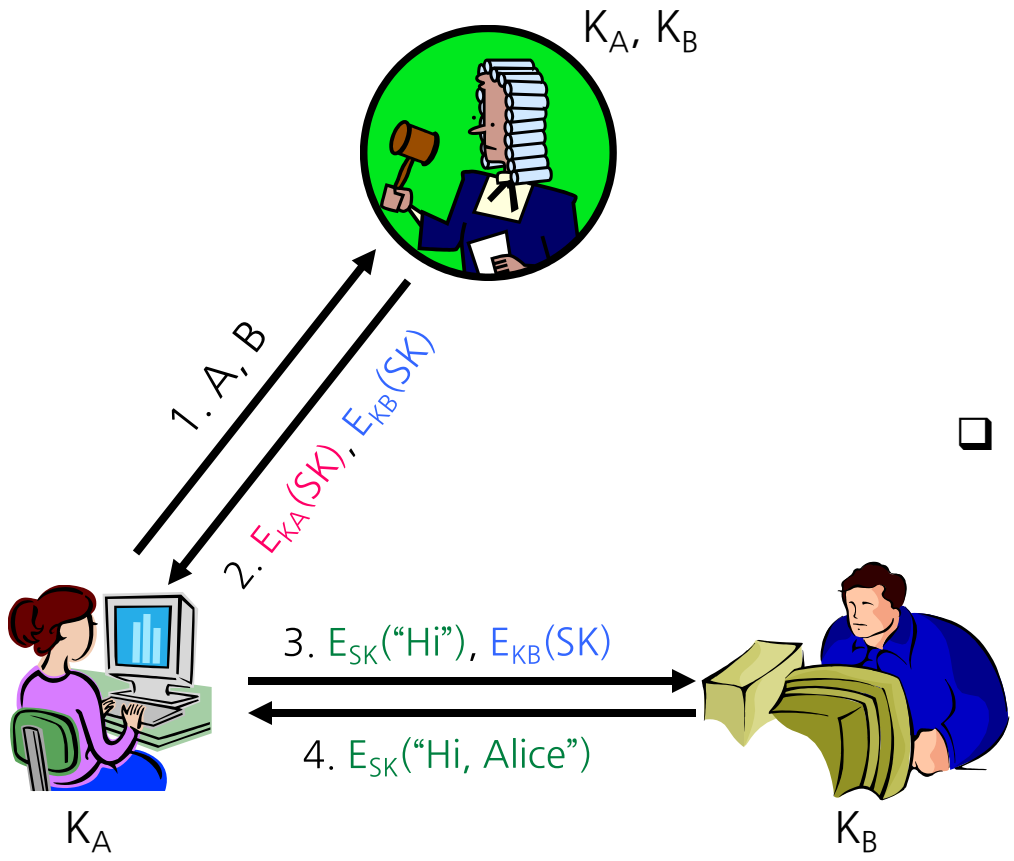
Yongdae Kim

Key Management

- Key establishment
  - Process to whereby a shared secret key becomes available to two or more parties
  - Subdivided into key agreement and key transport.

- Key management
  - The set of processes and mechanisms which support key establishment
  - The maintenance of ongoing keying relationships between parties

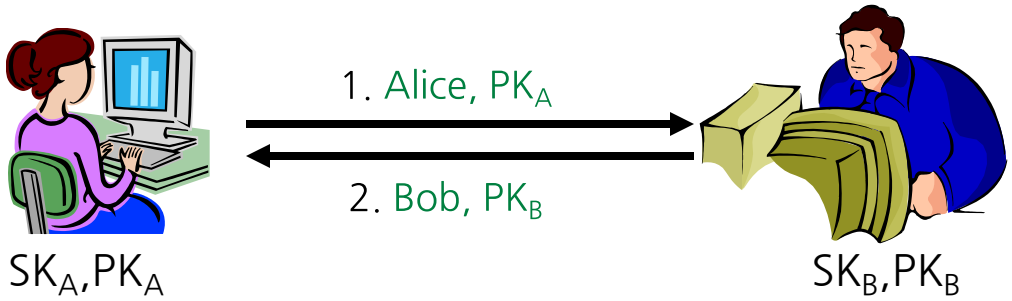
Key Management Through SKE



- Pros
  - Easy to add and remove entities
  - Each entity needs to store only one long-term secret key
- Cons
  - Initial interaction with the TTP
  - TTP needs to maintain *n* long-term secret keys
  - TTP can read all messages
  - Single point of failure

Key Management Through PKE

0xDAD12345	Alice
0xBADD00D1	Bob



- Advantages
  - TTP not required
  - Only *n* public keys need to be stored
  - The central repository could be a local file
- Problem
  - Public key authentication problem
- Solution
  - Need of TTP to certify the public key of each entity

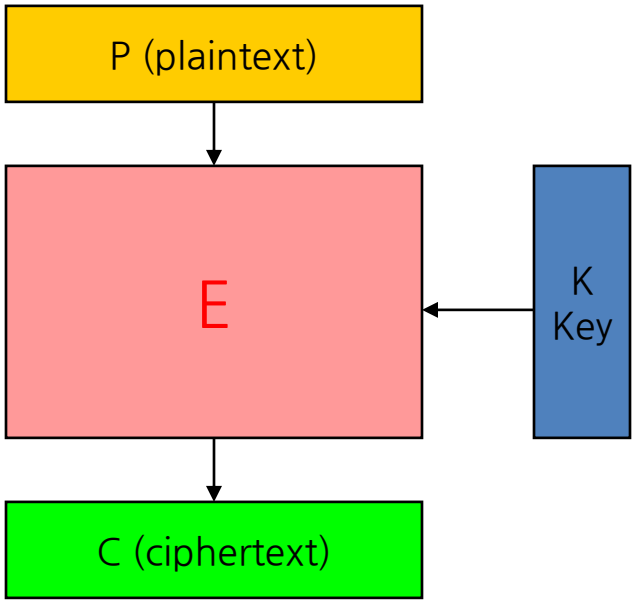
Kerckhoff's Principle

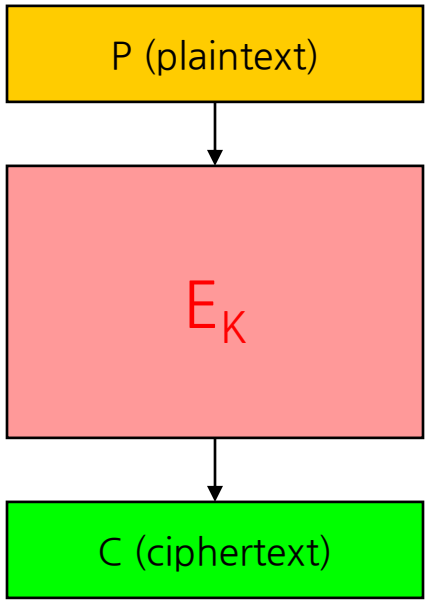
- Security should depend only on the key
  - Don't assume enemy won't know algorithm
    - » Can capture machines, disassemble programs, etc.
    - » Too expensive to invent new algorithm if it might have been compromised
  - Security through obscurity isn't
    - » Look at history of examples
    - » Better to have scrutiny by open experts

- "The enemy knows the system being used." (Claude Shannon)

Block Cipher

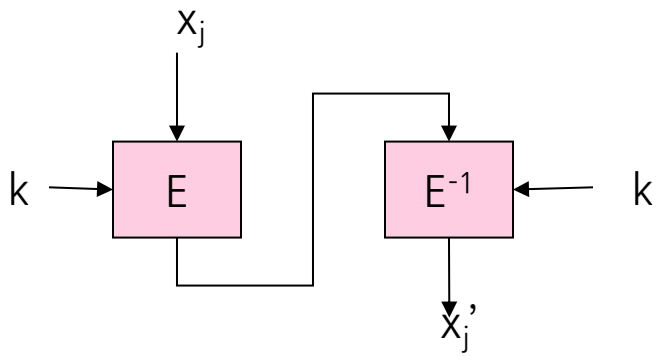
- E: $V_n \times K \rightarrow V_n$
  - $V_n = \{0,1\}^n$, $K = \{0, 1\}^k$, n is called block length, k is called key size
  - $E(P, K) = C$ for $K \in K$ and $P, C \in V_n$
  - $E(P, K) = E_K(P)$ is invertible mapping from V_n to V_n
    - » E_K: encryption function
  - $D(C, K) = D_K(C)$ is the inverse of E_K
    - » D_k: decryption function

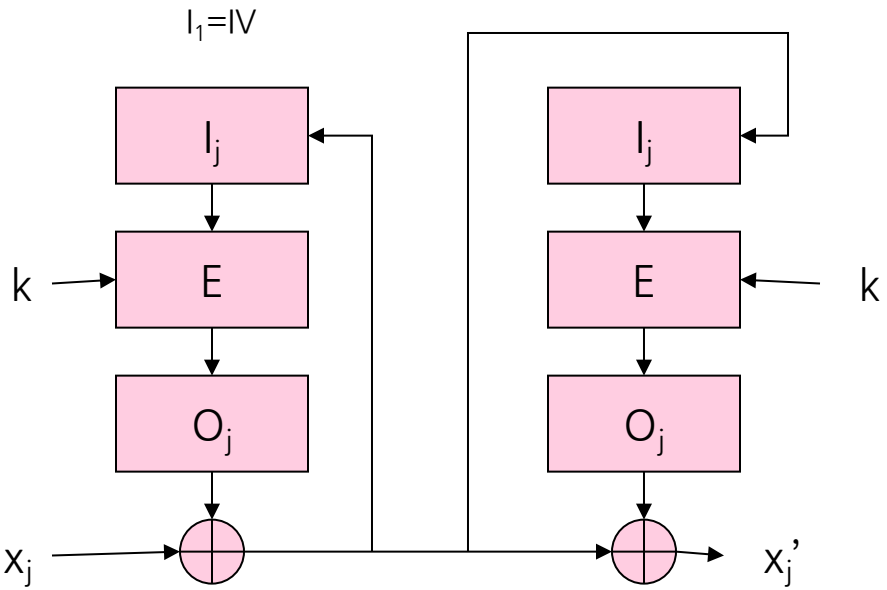


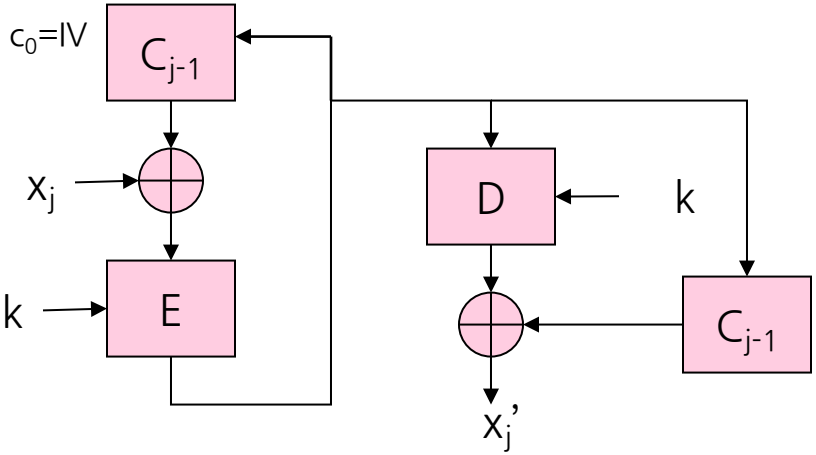


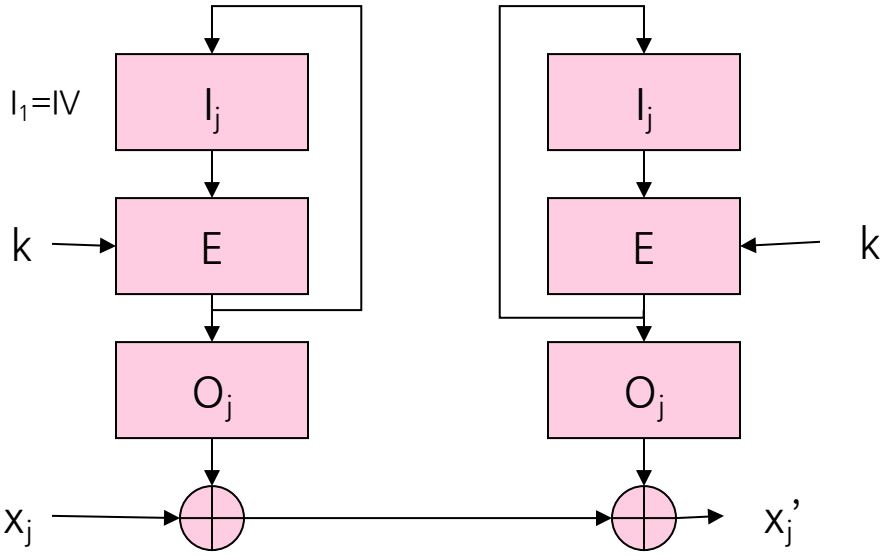
Modes of Operation

- A block cipher encrypts plaintext in fixed-size n-bit blocks (often n =128). What happens if your message is greater than the block size?





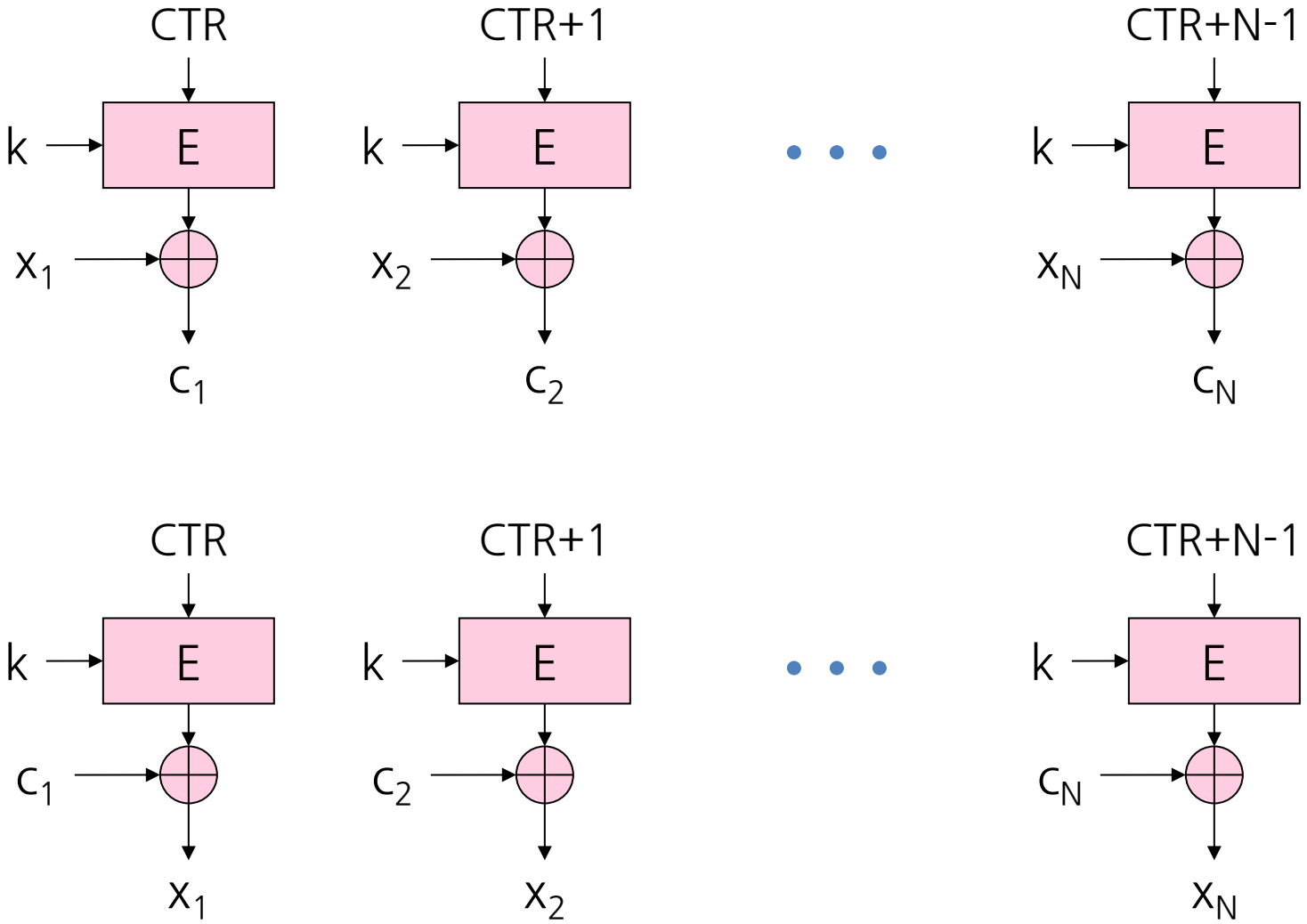




Modes of Operation

- ECB
  - Encryption: $c_j \leftarrow E_K(x_j)$
  - Decryption: $x_j \leftarrow E^{-1}{}_K(c_j)$
- CBC
  - Encryption: $c_0 \leftarrow IV$, $c_j \leftarrow E_K(c_{j-1} \oplus x_j)$
  - Decryption: $c_0 \leftarrow IV$, $x_j \leftarrow c_{j-1} \oplus E^{-1}{}_K(c_j)$
- CFB
  - Encryption: $I_1 \leftarrow IV$, $c_j \leftarrow x_j \oplus E_K(I_j)$, $I_{j+1} = c_j$
  - Decryption: $I_1 \leftarrow IV$, $x_j \leftarrow c_j \oplus E_K(I_j)$, $I_{j+1} = c_j$
- OFB
  - Encryption: $I_1 \leftarrow IV$, $o_j = E_K(I_j)$, $c_j \leftarrow x_j \oplus o_j$, $I_{j+1} = o_j$
  - Decryption: $I_1 \leftarrow IV$, $o_j = E_K(I_j)$, $x_j \leftarrow c_j \oplus o_j$, $I_{j+1} = o_j$

Modes of Operation (CTR)



CTR advantages

- Hardware efficiency
  - Parallelizable
- Software efficiency
  - Similar, modern processors support parallel computation
- Preprocessing
  - Pad can be computed earlier
- Random-access
  - Each ciphertext block can be encrypted independently
  - important in applications like hard-disk encryption
- Provable security
  - no worse than what one gets for CBC encryption
- Simplicity
  - No decryption algorithm and key scheduling

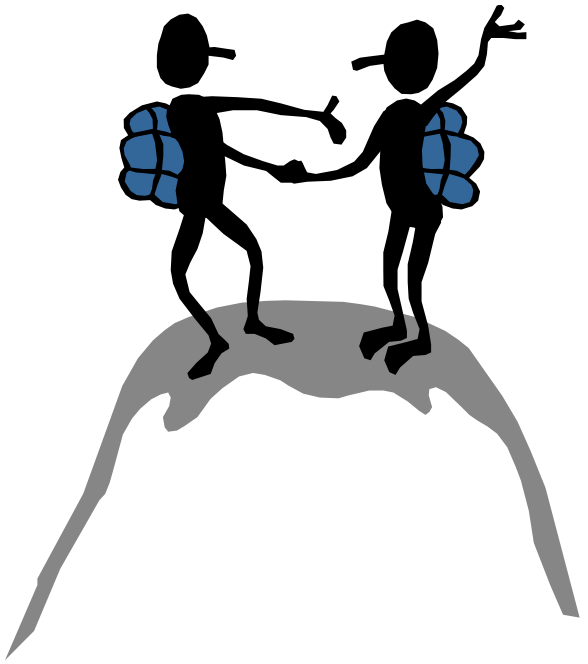
Double DES

- $C = E_{K2}[E_{K1}[P]]$
- $P = D_{K1}[D_{K2}[C]]$

- Reduction to single stage?
  - $E_{K2}[E_{K1}[P]] =? E_{K3}[P]$
  - It was proven that it does not hold

Meet-in-the-middle Attack

- Diffie 1977
- Exhaustively cracking it requires 2^{112}?
- $C = E_{K2}[E_{K1}\ [P]]$
  - $X = E_{K1}\ [P] = D_{K2}[C]$
- Given a known pair, (P, C)
  - Encrypt P with all possible 2^{56} values of K_1
  - Store this results and sort by X
  - Decrypt C with all possible 2^{56} K_2, and check table
  - If same, accept it as the correct key
- Are we done? &&#@!#(



Meet-in-the-middle Attack, cnt

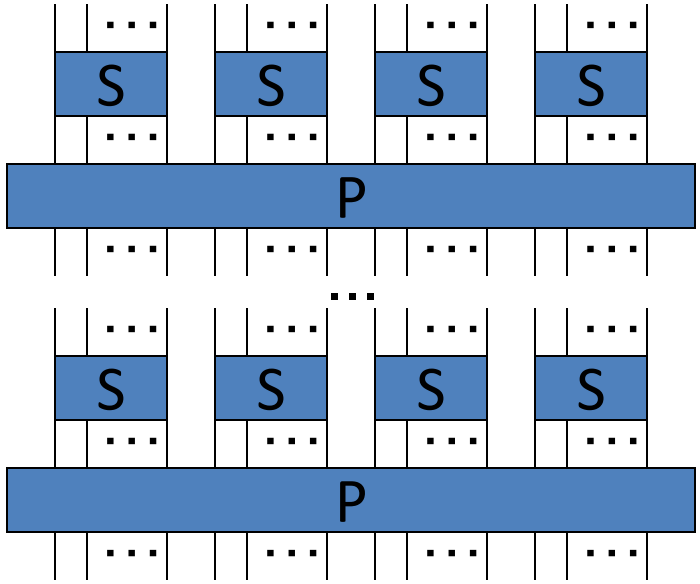
- Little statistics
  - For any P, there are 2^{64} possible C
  - DDES uses 112 bit key, so 2^{112} keys
  - Given C, there are $2^{112}/2^{64} = 2^{48}$ possible P
    - » So there are 2^{48} false alarms
  - If one more (P', C') pair, we can reduce it to 2^{-16}
- So using two (plaintext, ciphertext) pairs, we can break DDES c * 2^{56} encryption/decryption
- $C = E_{K2}[D_{K1}[P]]$ different?

Triple DES with two keys

- Obvious counter to DDES: Use three keys
  - Complexity?
  - 168 bit key
- Triple DES = EDE = encrypt-decrypt-encrypt
  - $C = E_{K1}[D_{K2}\ [E_{K1}[P]]]$
- Attacks?
  - No practical one so far

Product Cipher

- To build complex function to compose several simple operation offer complementary, but individually insufficient protection
- Basic operation: transposition, translation (XOR) and linear transformation, arithmetic operation, mod mult, simple substitution
- Substitution-permutation (SP) network is product cipher composed of a number of stages each involving substitution and permutation

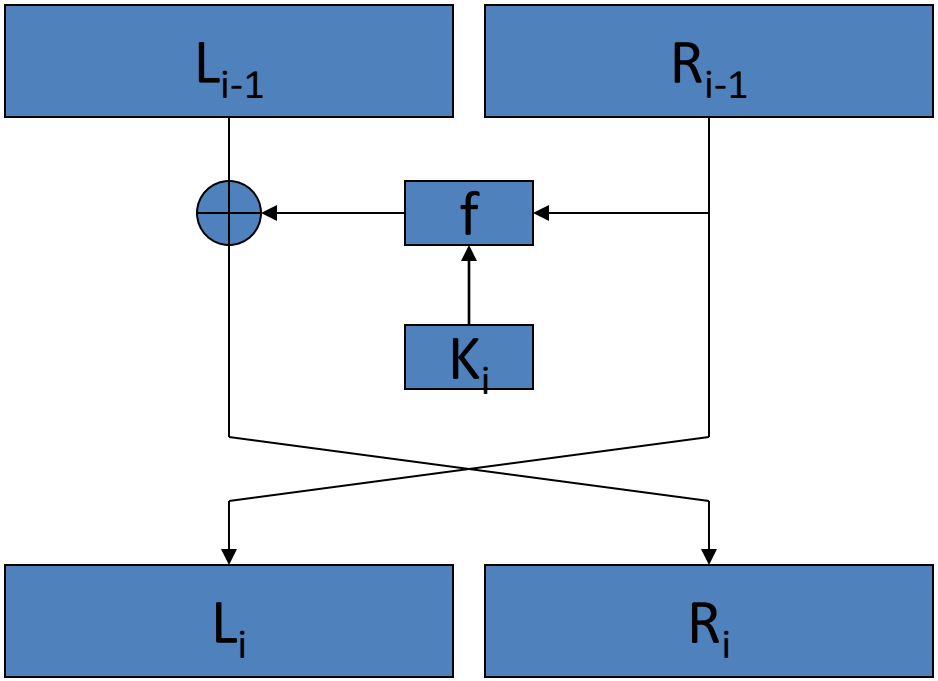


Feistel Cipher

- Virtually all conventional block ciphers
  - by Horst Feistel of IBM in 1973
- The realization of a Feistel Network depends on the choice of the following parameters and features:
  - Block size: larger block sizes mean greater security
  - Key Size: larger key size means greater security
  - Number of rounds: multiple rounds offer increasing security
  - Subkey generation algorithm: greater complexity will lead to greater difficulty of cryptanalysis.
  - Fast software encryption/decryption: the speed of execution of the algorithm becomes a concern

Feistel Network

- iterated cipher mapping (L_0, R_0) to (R_r, L_r) through r-round process, $(L_{i-1}, R_{i-1}) \rightarrow_{Ki} (L_i, R_i)$ as follows
  - $L_i = R_{i-1}$, $R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$, K_i is derived from K



Feistel Network - Why it works?

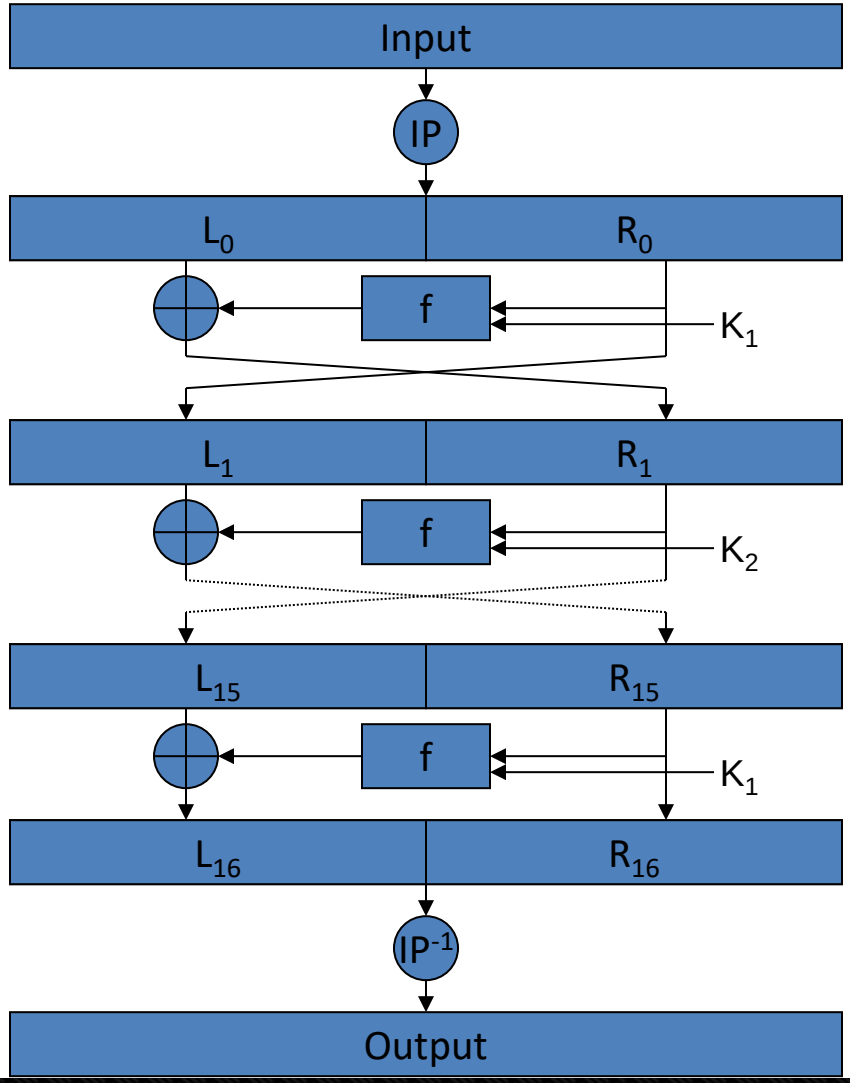
- 2 Round example
- Encryption
  - $L_1 = R_0$, $R_1 = L_0 \oplus f(K_1, R_0)$
  - $L_2 = R_1 = L_0 \oplus f(K_1, R_0)$, $R_2 = L_1 \oplus f(K_2, R_1)$
- Decryption
  - $R_1 = L_2$, $L_1 = R_2 \oplus f(K_2, R_1)$
  - $R_0 = L_1$, $L_0 = R_1 \oplus f(K_1, R_0)$
- Easily extensible to multi-round

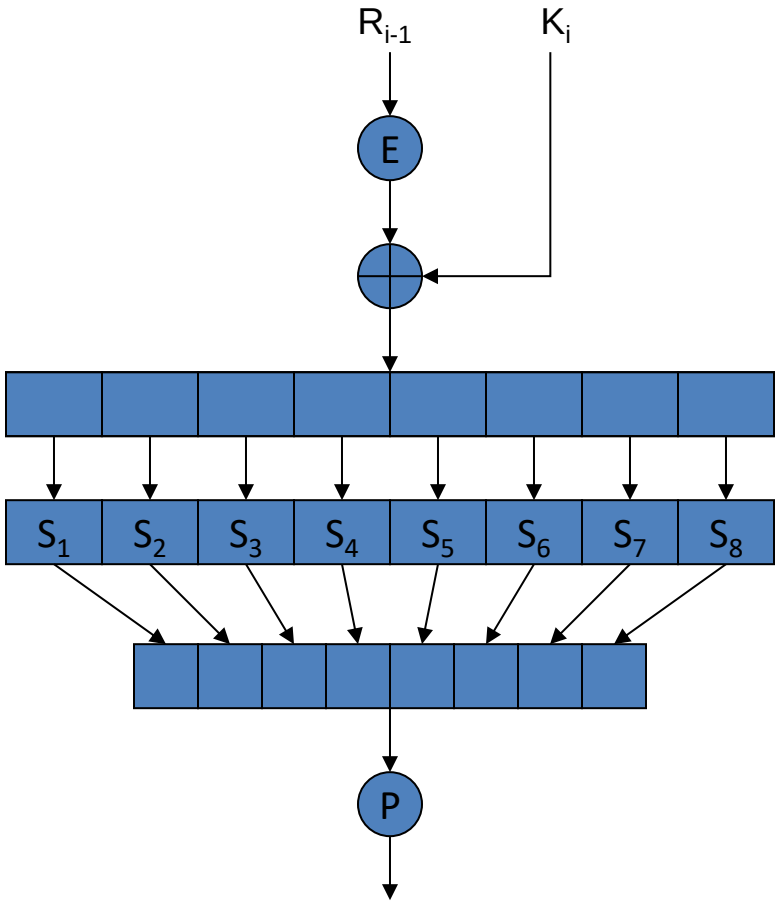
DES History

- Originated with early 1970's IBM effort to develop banking security systems
- First result was Lucifer, most common variant has 128-bit key and block size
  - Broken
- NBS (Currently NIST) called for Algorithms in 1973
- IBM submitted the best algorithm in 1977 and that became DES
  - Original IBM key size = 128, DES = 56 :-)
  - Design philosophy of S-Box was unknown
    - » Turned out to be strong

DES Overview

- |P|, |C| = 64, |K| = 56, 16 rounds, K ! sixteen 48-bit subkeys K_i are generated



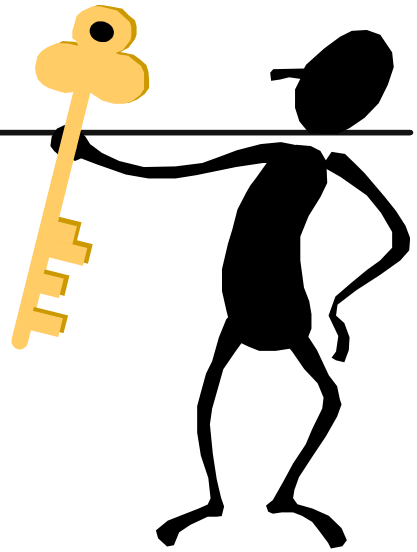


S-Box

- 6 bit input, 4 bit output
- 27 = **0**11011**1** = (**01**) (1101)
- S_1-Box output for 27 = 5

	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
1	0	15	7	4	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	3	10	5	0
3	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

New Era!



- DES broken
  - DES III Challenge by RSA
  - Idle CPU time of around 100,000 computers
  - In 22 hours
- Triple DES?
  - Original DES was designed for H/W implementation
  - 64 bit block size too small for security and efficiency
- Now what?

Advanced Encryption Standard

- In 1997, NIST issued a call for proposal
  - Block length = 128 bit
  - Key size = 128, 192, 256 bits
- In the first round, 15 algorithms were accepted
- Second round, 5 algorithms were selected
- In November 2001, final standard was published
  - Rijndel, FIPS PUB 197
  - http://csrc.nist.gov/publications/fips/fips197/fips-197.pdf
  - Joan Daemen and Vincent Rijmen

AES Evaluation Criteria

- Security
  - Actual security: compared with other submissions
  - Randomness: output is indistinguishable from random
  - Soundness: of mathematical basis
  - Other security factors: raised by security community
- Cost
  - No licensing: World-wide, non-exclusive, royalty-free
  - Computation efficiency: both S/W and H/W
  - Memory requirements
- Algorithm and Implementation characteristics
  - Flexibility: key-/block-size, wide variety of platforms
  - Simplicity

Stream Cipher

- Definition
  - encrypt individual characters of plaintext message one at a time, using encryption transformation which varies with time.
- Block vs. Stream
  - Block ciphers
    - » process plaintext in relatively large blocks
    - » The same function is used to encrypt successive blocks ⇒ memoryless
  - stream ciphers
    - » process plaintext in small blocks, and the encryption function may vary as plaintext is processed ⇒ have memory
    - » sometimes called state ciphers since encryption depends on not only the key and plaintext, but also on the current state.
  - This distinction between block and stream ciphers is not definitive
    - » adding memory to a block cipher (as in CBC) results in a stream cipher

One-time Pad and Stream Cipher

- One-time pad
  - Vernam cipher: $c_i = m_i \oplus x_i$ for i = 1, 2, 3…
  - 👍 key is generated independently and randomly
  - 👍 Ciphertext contributes no information about plain text
  - 👎 key should be as long as plaintext ⇒ key management

- Stream cipher tries to solve this problem having short key and generate pseudo-random sequence
  - Not unconditionally secure, but try to be computationally secure

Questions?

- Yongdae Kim
  - email: yongdaek@kaist.ac.kr
  - Home: http://syssec.kaist.ac.kr/~yongdaek
  - Facebook: https://www.facebook.com/y0ngdaek
  - Twitter: https://twitter.com/yongdaek
  - Google "Yongdae Kim"