Data Encryption Standard

Peter Chapin

Vermont State University

CIS-3240: Computer Security

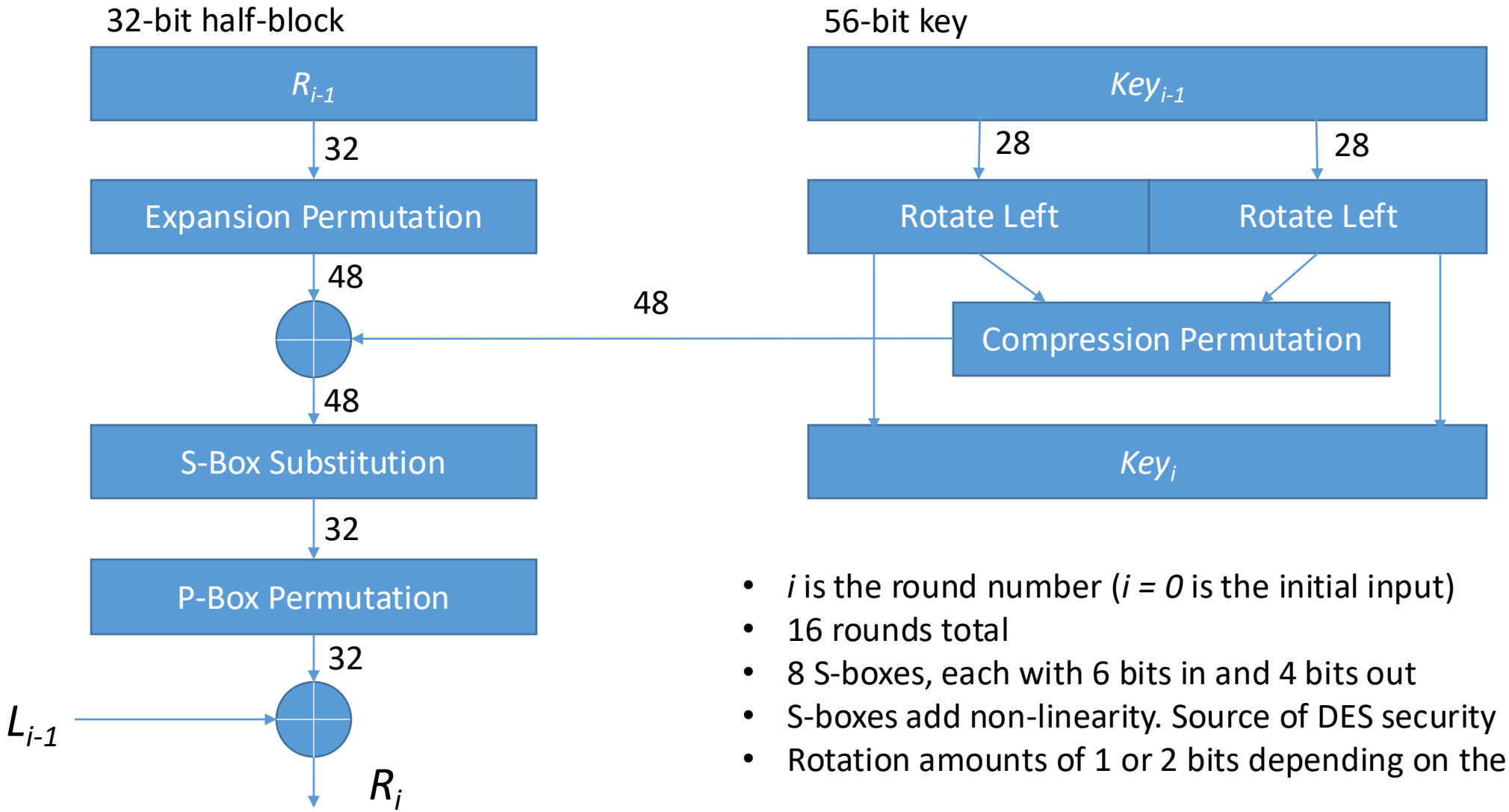
Brief History

- 1973: National Bureau of Standards (NBS) issued a request for design
- IBM submitted promising candidate based on Lucifer (an IBM cipher)
- NBS asked the NSA to evaluate the algorithm. IBM agreed to let others implement it (IBM had patented it)
  - The NSA made some changes but didn't explain why. Some people assumed they introduced a back door of some kind.
- 1975: NBS published the algorithm and asked for comments.
- 1976: DES adopted as standard: FIPS PUB 46 "Data Encryption Standard."

DES Basics

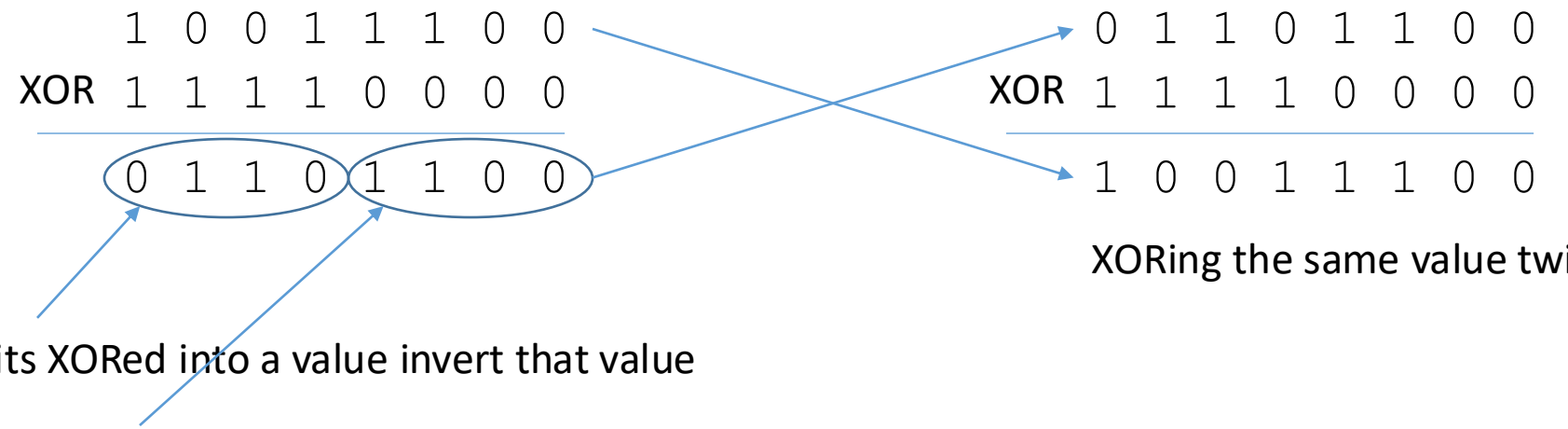
- 64-bit block cipher
- 56-bit key (considered too vulnerable to brute force today)
  - Key can be given as 8 ASCII characters (7 bits per character)
  - Key usually expressed as a 64-bit number with one parity bit ignored
- Feistel cipher
  - … with an initial permutation of the block and a final inverse permutation
- Fairly easy to implement in hardware
- Fairly awkward to implement in software

DES Function ‘f’



- *i* is the round number (*i = 0* is the initial input)
- 16 rounds total
- 8 S-boxes, each with 6 bits in and 4 bits out
- S-boxes add non-linearity. Source of DES security
- Rotation amounts of 1 or 2 bits depending on the round

Bitwise Exclusive OR (XOR)



XORing the same value twice recovers the original!

1 bits XORed into a value invert that value

0 bits XORed into a value keep that value

- A xor B = B xor A (communitive)
- (A xor B) xor C = A xor (B xor C) (associative)
- A xor 0 = A
- A xor A = 0
- (A xor B) xor A = (A xor A) xor B = 0 xor B = B

Weak Keys

- DES weak keys
  - `0x0000000, 0x0000000` <= 56 bits expressed as two 28-bit hex numbers
  - `0x0000000, 0xFFFFFFF`
  - `0xFFFFFFF, 0x0000000`
  - `0xFFFFFFF, 0xFFFFFFF`
- Rotations have no effect during subkey generation
- Most algorithms have some weak keys (not necessarily the same ones)
  - Programs can (and should) detect them and prevent them from being used

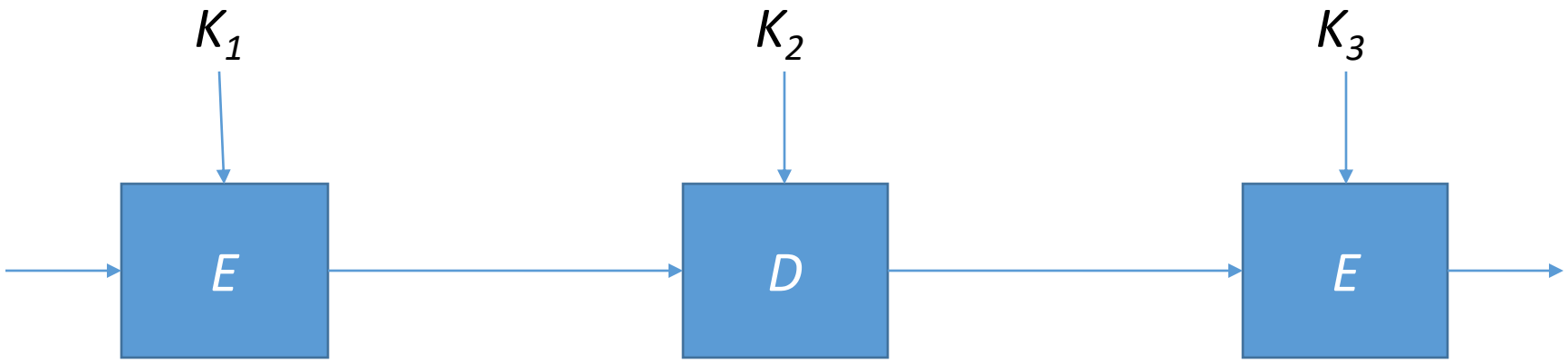
Workaround for Small Key: Encrypt Twice?

- Double DES… using two different keys, K_1 and K_2
- But wait!
  - $E_{K2}(E_{K1}(P)) = E_{K3}(P)$ for some K_3?
  - If so, Double DES would offer no additional security.
  - Encryption with a key is like a permutation of plaintext blocks to ciphertext
  - Application of two permutations will produce another
  - Question: Is DES closed (will two DES permutations produce another in DES)?
- 1992: It was proved that DES is not closed
- Unfortunately, Double DES is not significantly better than single DES.
  - Not worth the trouble

Meet In The Middle Attack

- General method of attack in cases like this…
  - Assume you have a known (plaintext, ciphertext) pair
    - Ciphertext result from plaintext you know or can guess
  1. Compute table of all possible encryptions (2^{56} of them)
  2. Compute all decryptions of ciphertext
     1. For each decryption, see if the encryption is in the table
     2. If so, you have found the two keys
  - Only requires 2^{57} operations
  - BUT… does require storage of 2^{56} ciphertext blocks
    - $2^{56} * 8 = 2^{59}$ bytes = 2^{47} Terabytes
    - Could be a problem ☺

Triple DES (3DES)



- Interoperability with single DES: Let $K_1 = K_2 = K_3$
- Usually, 3DES is used with two keys: $K_1 = K_3$ and K_2 giving 112 bits of key material
- Notice 3DES with two keys is no faster than 3DES with three keys.
- 3DES is still used, but it tends to be slow.