Confidentiality

It's just between you and me

Symmetric Encryption

- Both parties, Alice and Bob, share a secret key *K*
- Given a message *M* and a key *K*
  - *M* is known as the plaintext or cleartext, what we're trying to keep secret
  - Encrypt: *E(K, M)* → *C*, C is known as the ciphertext
  - Decrypt: *D(K, C)* → *M*
  - Attacker cannot derive *M* from *C* without *K* (or trying every *K*...)

- *E* and *D* take the same key *K*, hence the name "symmetric" encryption
- Examples: AES, Blowfish, DES, RC4

One-Time Pad

- Generate a perfectly random stream of bits K
- $E(K, M) = M \oplus K$
- $D(K, C) = C \oplus K \quad (K \oplus K == 0)$
- "Perfect" secrecy
  - Informationally-theoretic secure: given C but not K, M could be anything!
  - Fast: XORing is cheap and fast
- Totally impractical
  - Need a very big key, same size as all data

Quiz: Two Time Pad

- You generate a perfectly random key K. You send two messages, M_1 and M_2, encrypted with K ($C_1 = K \oplus M_1$, $C_2 = K \oplus M_2$). Does sending C_2 leak information about M_1 and M_2?

Quiz: Two Time Pad

- You generate a perfectly random key K. You send two messages, M_1 and M_2, encrypted with K ($C_1 = K \oplus M_1$, $C_2 = K \oplus M_2$). Does sending C_2 leak information about M_1 and M_2?

  Yes. If an adversary Eve hears both C_1 and C_2 , she can reconstruct $M_1 \oplus M_2$. Recall that $C_1 = K \oplus M_1$, $C_2 = K \oplus M_2$. Therefore,

  $$C_1 \oplus C_2 \quad = \quad (K \oplus M_1) \oplus (K \oplus M_2) \quad = \quad M_1 \oplus M_2$$

  For example, if $M_1 = M_2$, then $C_1 = C_2$.

  Called a "one time pad" for a reason!

Idea: Computational Security

- Distribute small *K* (e.g., 128 bits, 256 bits) securely
- Use *K* to encrypt much larger *M*
- Given C = *E(K, M)*, may be only one possible *M*
  - If *M* has redundancy
- But believed computationally intractable to find
  - Could try all possible *K*, but 2^{128} is a lot of work!

Ciphers

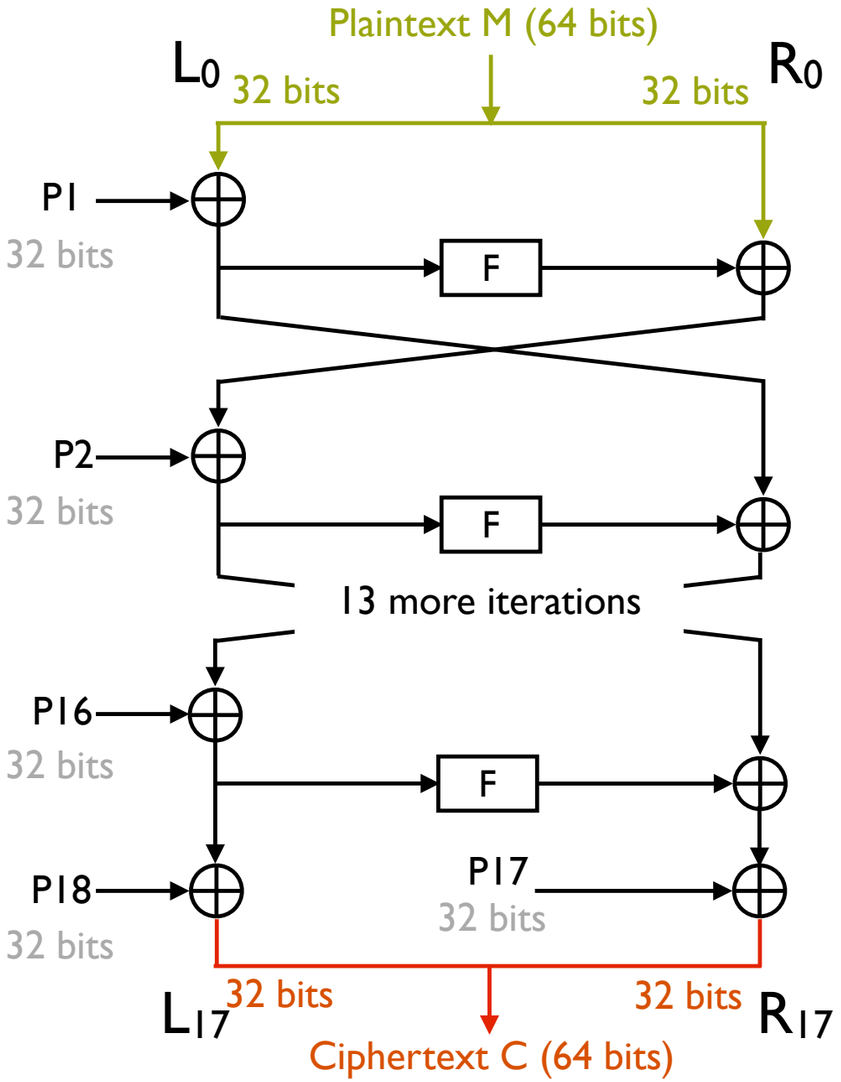
- Stream ciphers: pseudo-random pad
  - Generate a pseudo-random sequence of bits based on key
  - Encrypt/decrypt by XORing like one-time pad
  - BUT NOT A ONE-TIME PAD! Immediately mistrust anyone who says so!
  - Have run into many problems in practice, so I'd recommend avoiding them
    - Example: 802.11 WEP shown broken in ~2001, replaced by WPA in 2003, WPA2 in 2004
- Block ciphers
  - Operate on fixed sized blocks (64 bits, 128 bits, etc.)
  - Maps plaintext blocks to ciphertext blocks
  - Today, should use generally AES: many other algorithms

Ciphers

- Stream ciphers: pseudo-random pad
  - Generate a pseudo-random sequence of bits based on key
  -
  -
  -
- B
  -
  - Maps plaintext blocks to ciphertext blocks
  - Today, should use generally AES: many other algorithms

WARNING: just giving a feel on how algorithms work, many details missing! Don't try it this at home!

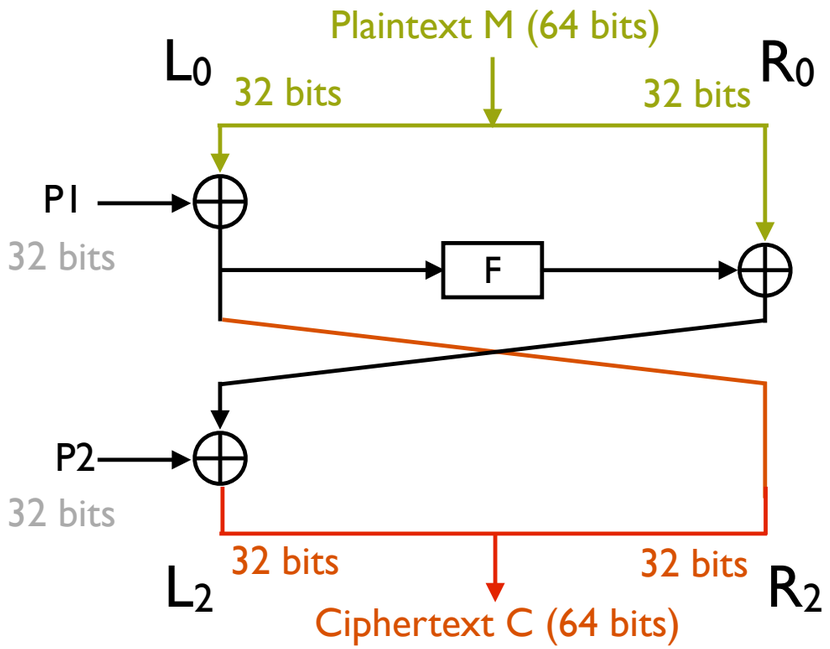
Example Block Cipher: Blowfish



- Derive F and 18 subkeys (P_1 ... P_{18}) from key
- Divide plaintext block into two halves, L_0 and R_0
  - $R_i = L_{i-1} \oplus P_i$
  - $L_i = R_{i-1} \oplus F(R_i)$
  - $R_{17} = L_{16} \oplus P_{17}$
  - $L_{17} = R_{16} \oplus P_{18}$
  - Output $L_{17}R_{17}$

- Not a complete description!

Block Cipher Quiz



I didn't take CS144, so I decide Blowfish is too slow and cut it to one iteration as on the left. I feed it two plaintext blocks. L_0 and L_2 are underlined.

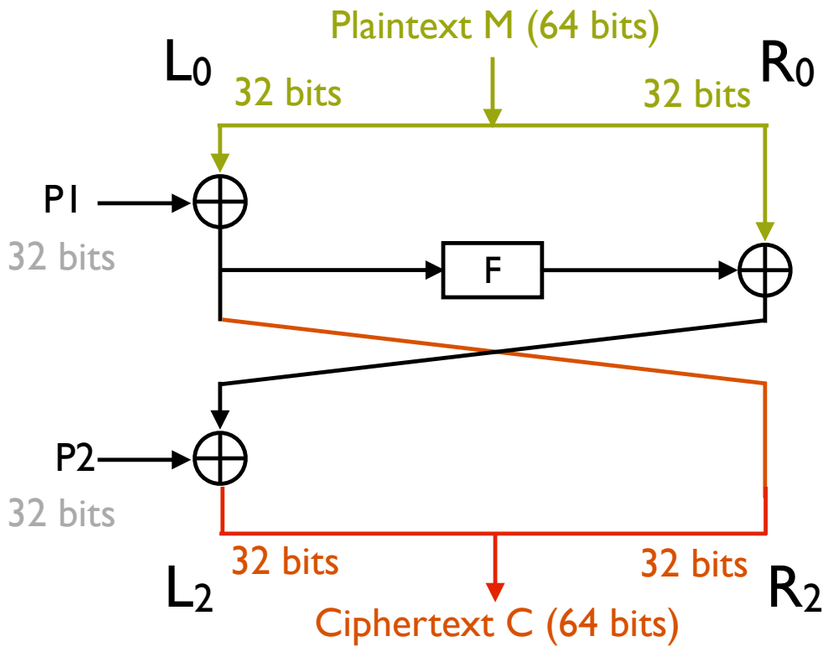
The first plaintext block is `0x00000000`00000000
producing ciphertext `0x4F072DBC`7B77DCA2

The second ciphertext is `0x4F072DBC`90DCDC85

What is the value of L0 in the second plaintext block?

a. EBAB0027
b. 5414D8F7
c. AA0A0072
d. E6A30055
e. E6A30057
f. impossible to determine

Block Cipher Quiz



I didn't take CS144, so I decide Blowfish is too slow and cut it to one iteration as on the left. I feed it two plaintext blocks. L_0 and L_2 are underlined.

The first plaintext block is `0x00000000`00000000
producing ciphertext `0x4F072DBC`7B77DCA2

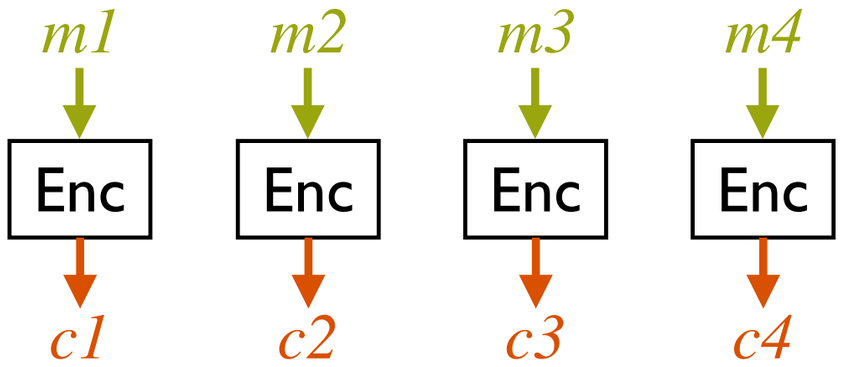
The second ciphertext is `0x4F072DBC`90DCDC85

What is the value of L0 in the second plaintext block?

a. 0xEBAB0027
b. 0x5414D8F7
c. 0xAA0A0072
d. 0xE6A30025
e. 0xE6A30057
f. impossible to determine

Using Block Ciphers

- Messages are typically longer than one block
- ECB (electronic code book) mode
  - Break plaintext into blocks, encrypt each block separately

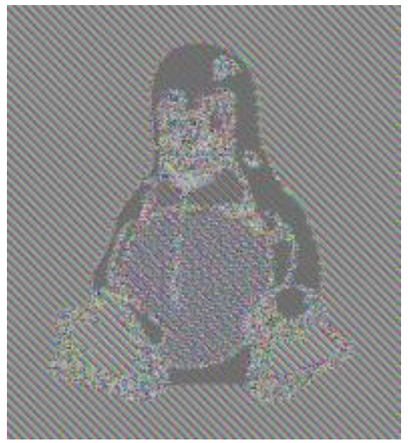


  - Attacker can't decrypt any of the blocks; message secure
  - Can re-use keys, every block encrypted with cipher will be secure

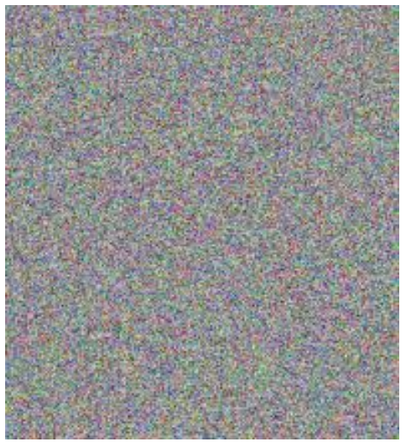
WRONG!

- Attacker will learn of repeated plaintext blocks
  - If transmitting a sparse file, will know where non-zero regions lie
- Example: military instructions
  - Most days, send encrypted "nothing to report"
  - One day, send "attack today", attacker will know plans are being made

source



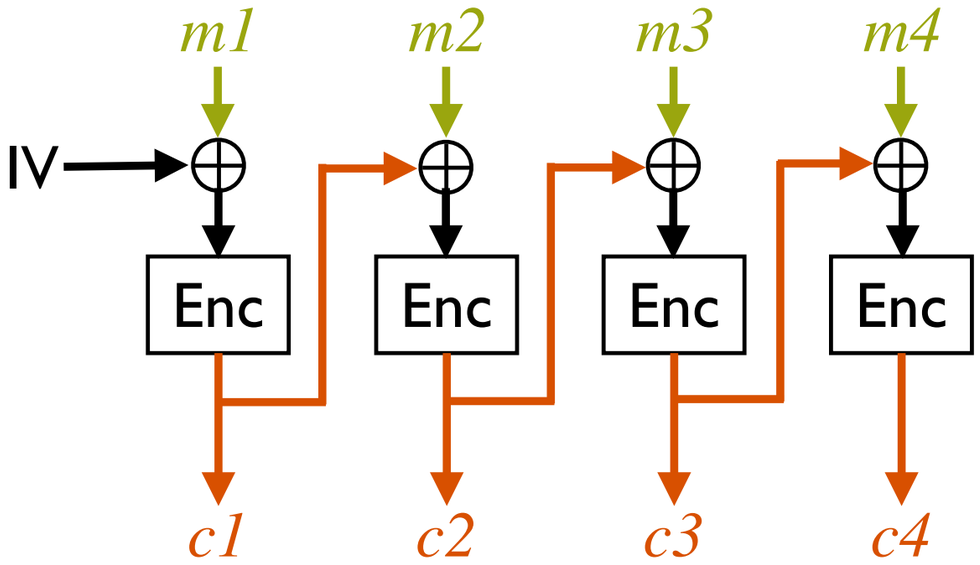
ECB



desired output

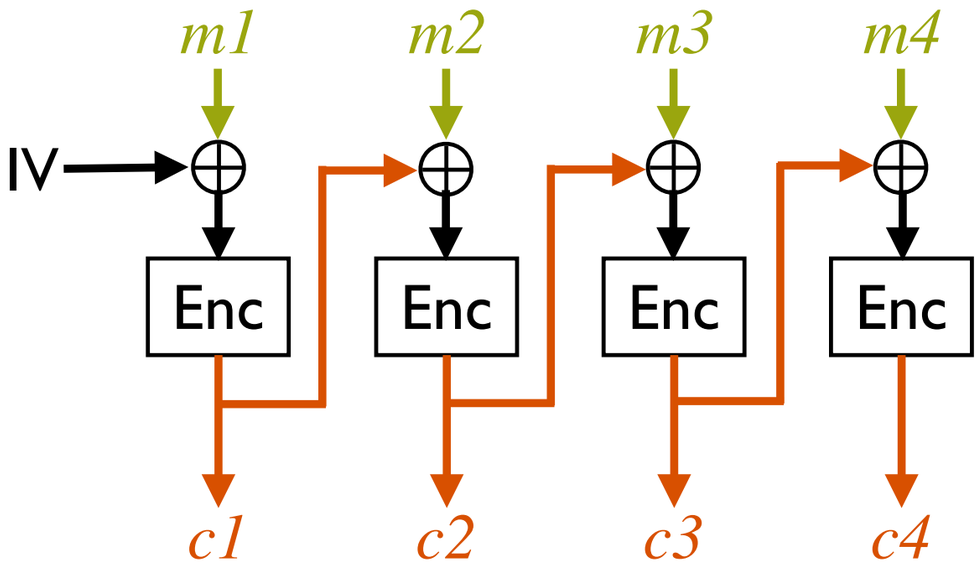
http://en.wikipedia.org/wiki/Block_cipher_mode_of_operation

Cipher Block Chaining (CBC) Mode

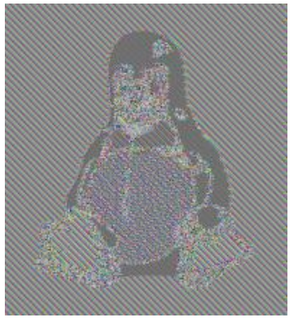


- Choose initialization vector (IV) for each message, can be publicly known
  - Can be 0 iff key only ever used to encrypt one message
  - Choose randomly for each message if key re-used
- $C_1 = E(K, M_1 \oplus IV)$, $C_i = E(K, M_i \oplus C_{i-1})$

Cipher Block Chaining (CBC) Mode



source



ECB

CBC

- Choose initialization vector (IV) for each message, can be publicly known
  - Can be 0 iff key only ever used to encrypt one message
  - Choose randomly for each message if key re-used
- $C_1 = E(K, M_1 \oplus IV), C_i = E(K, M_i \oplus C_{i-1})$

Many Other Modes!

- Cipher Feedback (CFB) mode: $C_i = M_i \oplus E(K, C_{i-1})$
  - Useful for message that are not multiple of block size
- Output Feedback (OFB) mode
  - Repeatedly encrypt IV & use result like stream cipher
- Counter (CTR) mode: $C_i = M_i \oplus E(K, i)$
  - Useful if you want to encrypt in parallel

- Quiz: given a shared secret key, can you transmit files secured from adversaries over a network solely by encrypting them in CBC mode?

Many Other Modes!

- Cipher Feedback (CFB) mode: $C_i = M_i \oplus E(K, C_{i-1})$
  - Useful for message that are not multiple of block size
- Output Feedback (OFB) mode
  - Repeatedly encrypt IV & use result like stream cipher
- Counter (CTR) mode: $C_i = M_i \oplus E(K, i)$
  - Useful if you want to encrypt in parallel

- Quiz: given a shared secret key, can you transmit files secured from adversaries over a network solely by encrypting them in CBC mode?