TOPIC: 6.857 - Computer & Network Security

DATE: 2/22/12

FILE UNDER:

PAGE: L5.1

Admin:

Pset #1 due Friday
Email to 6857-staff@mit.edu

Guest lecturers next week: Mon: Victor Costan web security
Wed: Srini Devadas authentication & PUF's

Today:

DES (Data Encryption Standard)
AES (Advanced Encryption Standard)
Ideal block cipher defn.
Modes of operation:
- ECB
- CTR
- CBC
- CFB

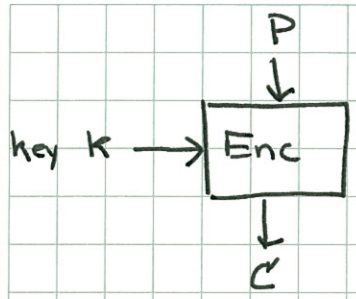
IND-CCA security defn
UFE mode

TOPIC:

DATE:

FILE UNDER:

Block ciphers:



plaintext block

ciphertext block

fixed-length P, C, K

DES: $|P| = |C| = 64$ bits $|K| = 56$ bits

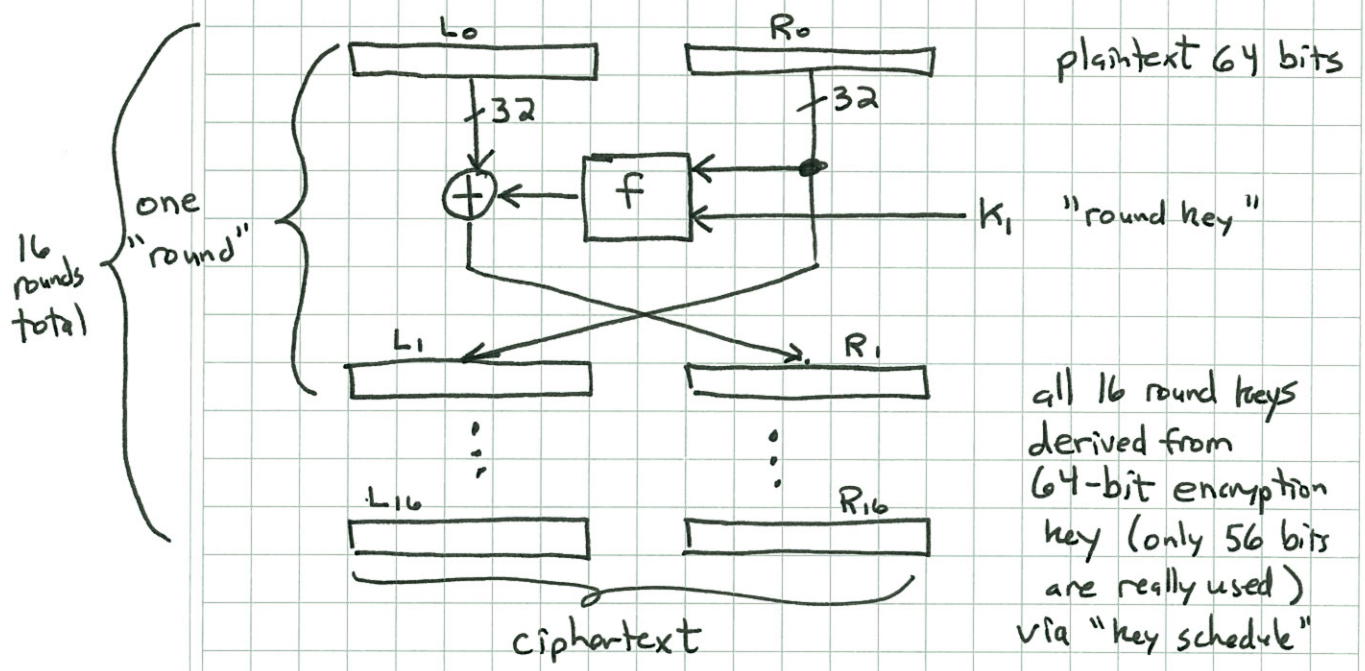
AES: $|P| = |C| = 128$ bits $|K| = 128, 192, 256$ bits

Use a "mode of operation" to handle variable-length input.

DES

"Data Encryption Standard"
Standarized in 1976. Now deprecated in favor of AES.

"Feistel structure":

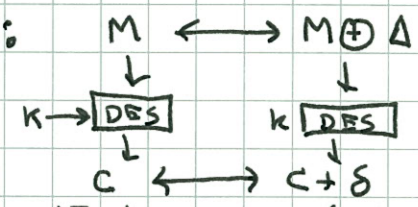


Note: Invertible for any f and any key schedule.

f uses 8 "S-boxes" mapping 6-bits ⇒ 4 bits nonlinearly.

Key is too short! (Breakable now quite easily by brute-force)

Subject to differential attacks:



2^{47} chosen pairs (Biham/Shamir)

Subject to linear attacks:

e.g. if $M_3 \oplus M_{15} \oplus C_2 \oplus K_{14} = 0$ (eqn on bits)
with prob $p = 1/2 + \varepsilon$

then need $1/\varepsilon^2$ samples to break (Matsui, 2^{43} PT/CT pairs)

TOPIC:

DATE:

FILE UNDER:

AES

"Advanced Encryption Standard" (U.S. govt)

Replaces DES

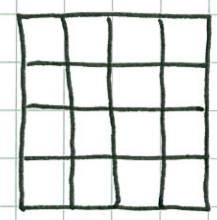
AES "contest" 1997–1999:
15 algorithms submitted: RC6, Mars, Twofish, Rijndael, ...
winner = Rijndael (by Joan Daemen & Vincent Rijmen, (Belgians))

Specs: 128-bit plaintext/ciphertext blocks
128, 192, or 256-bit key
10, 12, or 14 rounds (dep. on key length)

Byte-oriented design (some math done in Galois field $GF(2^8)$)

View input as 4x4 byte array:

$4 \times 4 \times 8 = 128$



For version with 128-bit keys, 10 rounds:

- Derive 11 "round keys", each 128 bits (4x4x byte)
- In each round:
  1. XOR round key
  2. Substitute bytes (lookup table)
  3. Rotate rows (by different amts)
  4. Mix each column (by linear opn)
- Output final state

See readings for details.
There are very fast implementations. Also Intel has put supporting hardware into its CPU's.

Security: Good; perhaps # rounds should be a bit larger...

For practical purposes, can treat AES as <u>ideal block cipher</u>:

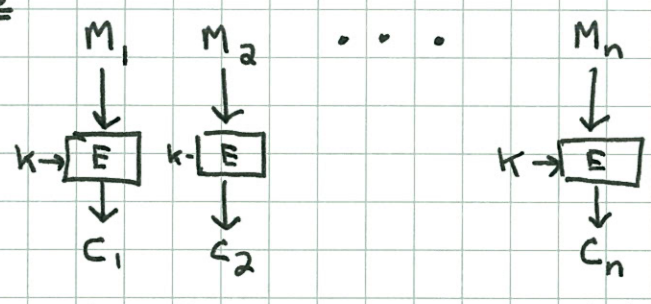
[For each key, mapping $Enc(K, \cdot)$ is a random independent permutation of $\{0,1\}^{128}$ to itself.

Modes of Operation:

How to encrypt variable-length messages? (using AES)

"ECB" = "Electronic code book"
"CTR" = "Counter mode"
"CBC" = "Cipher-block chaining" (& CBC-MAC)
"CFB" = "Cipher feedback"
... (others...)

ECB:



← divide data into b-bit blocks, where b = input block size

To handle data that is not a multiple of b bits in length:
- Append a "1" bit (<u>always</u>)
- Append enough "0" bits to make length a multiple of b bits.

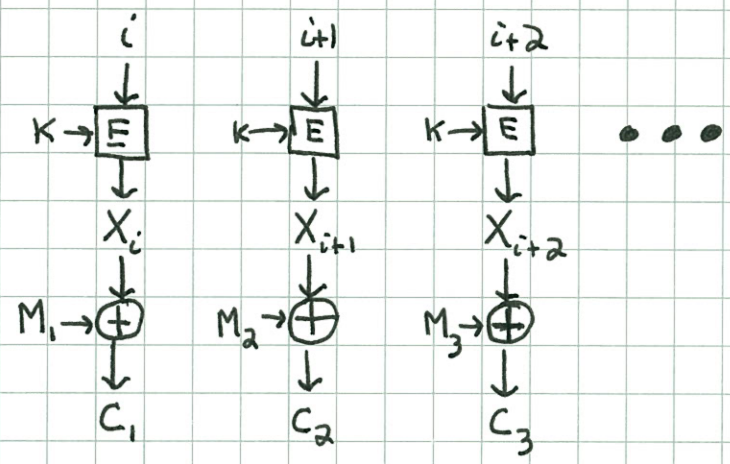
This gives <u>invertible</u> (1-1) "padding" operation.
Pad before encryption; unpad after decryption.

ECB preserves many patterns: repeated message blocks
⇒ repeated ciphertext blocks

ECB really only good for encrypting <u>random</u> data (e.g. keys)

CTR (Counter mode):

Generate a PR (pseudorandom) sequence by encrypting $i, i+1, \ldots$
XOR with message to obtain ciphertext.



Initial counter value can be transmitted first:

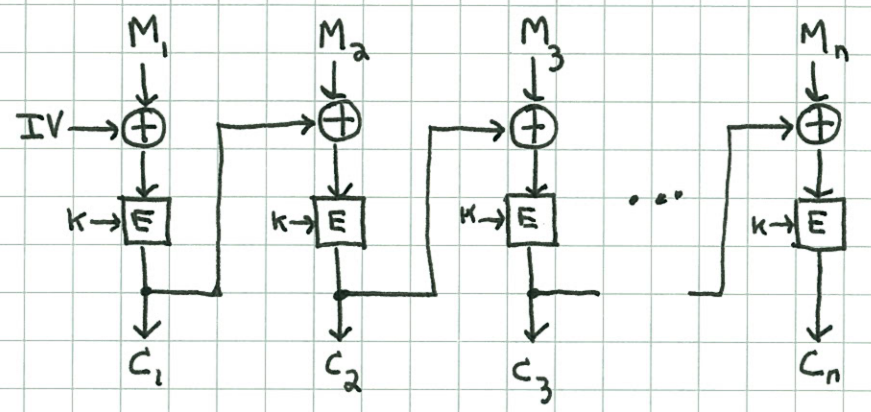
$$i, C_1, C_2, \ldots$$

Of course, no counter value should be re-used!

CBC (Cipher-block chaining):

Choose IV ("initialization value") randomly, then use each C_i is "IV" for M_{i+1}. Transmit IV with ciphertext:

$$IV, C_1, C_2, \ldots, C_n$$



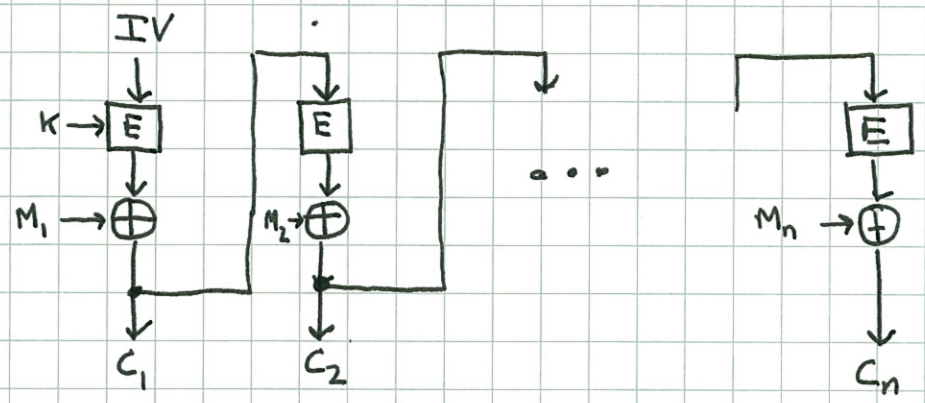
Decryption easy, and parallelizable (∴ little error propagation)

Lookup "ciphertext stealing" for cute way of handling messages that are not a multiple of b bits in length. This method give ciphertext length = message length.

Last block C_n is the "CBC-MAC" (CBC Message Authentication code) for message M. [A <u>fixed</u> IV is used here.] The MAC is a "cryptographic checksum" (more later...) (If messages have variable length then key for last block should be different.)

CFB (Cipher feedback mode)

Similar to CBC mode. Uses random IV transmitted with ciphertext.



If M is not a multiple of b bits in length, can just transmit shortened ciphertext. (No need for ciphertext stealing.)

TOPIC:

DATE:

FILE UNDER:

Are these modes good ones? What do we want?

If block cipher is indistinguishable from ideal block cipher then mode provides indistinguishability based on chosen ciphertext attack (IND-CCA):

- Define as game with adversary.
- Mode is IND-CCA secure if adversary can win with probability at most $1/2 + \varepsilon$ for "negligible" ε.

Let K be randomly chosen key.
Let E_K denote encryption (using mode) with key K.
Let D_K denote decryption

<u>Phase I ("Find"):</u>

- Adversary given black-box access to E_K, D_K (can encrypt/decrypt whatever it likes)
- Adversary outputs two messages m_0, m_1 of same length, plus state information s.

<u>Phase II ("Guess"):</u>

- Examiner secretly picks $d \xleftarrow{R} \{0,1\}$
  Examiner computes $y = E_K(m_d)$
- Adversary given y, s, access to E_K, and access to D_K (<u>except</u> on y)
- Adversary computes for a while, then must produce bit $\hat{d}$ as its guess for d.
- Adversary's advantage is $\left| P(\hat{d}=d) - 1/2 \right|$.

Encryption secure against CCA attack if advantage is negligible.

TOPIC:

DATE:

FILE UNDER:

Fact: To be IND-CCA secure, method must be randomized. !

(else Adv can encrypt m_0, m_1 & compare to y)

& randomization should not be evident to Adv (i.e. not usable for decryption)

Previous modes are not IND-CCA secure!

ECB: not randomized

CTR: starting counter value might be random, but it is transmitted in clear.
In any case, it is legal for Adv to ask for decryption of prefix of y (giving prefix of m_d)

CBC: similar to CTR: IV might be random, but it is xmitted in clear.
Decryption of prefix of y also works.

CFB: Same. IV in clear; prefix argument works.

Can one design a IND-CCA scheme?

TOPIC:

DATE:

FILE UNDER:

Here is a <u>sketch</u> of one IND-CCA secure method.
(due to Desai. UFE = "Unbalanced Feistel encryption")

M = long message, sequence $M_1, M_2, \ldots, M_n$ of b-bit blocks.

$K = (K_1, K_2, K_3)$ Three indep. keys for block ciphers

$r \xleftarrow{R} \{0,1\}^b$ starting counter value

<u>pad</u> $P = P_1, P_2, \ldots, P_n$ where $P_i = E_{K_1}(r+i)$ ← (CTR mode)

<u>ciphertext</u> $C = C_1, C_2, \ldots, C_n$ where $C_i = M_i \oplus P_i$

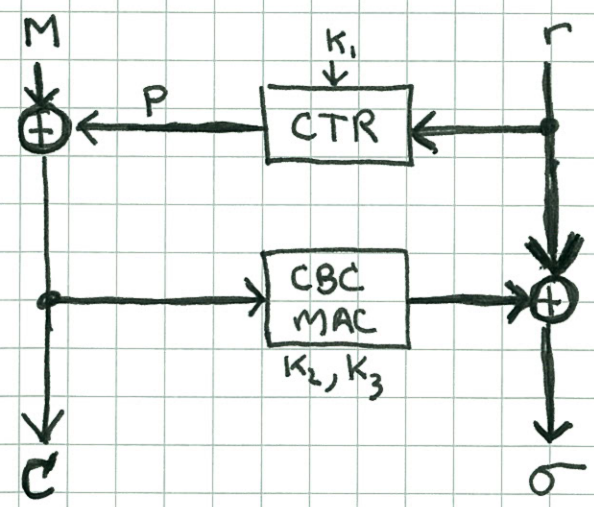
CBC-MAC: $X_0 = 0^b$

$X_i = E_{K_2}(X_{i-1} \oplus C_i) \quad 1 \le i < n$

$X_n = E_{K_3}(X_{n-1} \oplus C_n)$ (MAC)

$\sigma = r \oplus X_n$ use MAC to mask r

<u>Output:</u> $C_1, C_2, \ldots, C_n, \sigma$



TOPIC:

DATE:

FILE UNDER:

- Encryption with UFE can be done in single pass over data, but decryption requires two passes:
  - first to compute max X_n, then to get r
  - second to decrypt C to get M
- Only designed for <u>confidentiality</u> (there is no way provided for receiver to tell if ciphertext has been tampered with.) (Need to use MAC on top of all of this, or some "combined mode" providing <u>both</u> confidentiality & integrity.)
- Note "unbalanced Feistel structure".
- Length of ciphertext $(C, \sigma) = |M| + |r|$;
  expansion only as needed for randomization.
  No need for "ciphertext stealing" since we use CTR mode.